

T.C.
GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ
BİLGİSAYAR BİLİMLERİ
VERİ TABANI YÖNETİM SİSTEMLERİ

SQL ile VERİ TABANI İŞLEMLERİ

İshak KUTLU

Mayıs, 2023

Ankara

İçindekiler

MS Excel'den MS SQL Server'a Tablo Aktarımı	2
1. SELECT İfadesi, WHERE Yapısı, AND, OR, NOT Operatörleri (3 puan)	2
2. DISTINCT (3 puan)	3
3. JOIN (LEFT JOIN, RIGHT JOIN, INNER JOIN) (7 puan)	4
4. ORDER BY (3 puan)	7
5. GROUP BY (3 puan)	8
6. HAVING (3 puan).....	9
7. IN ve LIKE (3 puan)	10
8. BETWEEN (3 puan)	12
9. UNION (3 puan).....	13
10. COUNT, SUM Fonksiyonları (3 puan).....	14
11. MAX, MIN, AVG Fonksiyonları (3 puan)	16
12. DATEDIFF ve DATEADD Fonksiyonları (3 puan).....	19
13. İç İç SELECT İfadesi (Nested Query) (6 puan)	21
14. Foreign Key (Yabancı Anahtar) Tanımlama (3 puan)	25
15. INSERT İfadesi (3 puan).....	26
16. INSERT INTO-SELECT İfadesi (3 Puan).....	27
17. UPDATE İfadesi (3 puan).....	29
18. DELETE İfadesi (3 puan)	29
19. CREATE TABLE (3 puan)	30
20. ALTER TABLE (3 puan).....	31
21. DROP TABLE İfadeleri (3 puan)	33
22. CREATE PROCEDURE (İşlevi makul, mantıklı ve yeterli olacaktır. IF yapısı, CURSOR yapısı, Hata denetimi, transaction, eş zaman kontrolü içerecektir) (10 puan)	34
23. CREATE TRIGGER (İşlevi makul, mantıklı ve yeterli olacaktır, 7 puan)	40
24. CREATE VIEW (3 puan)	42
25. CREATE FUNCTION (İşlevi makul, mantıklı ve yeterli olacaktır, 7 puan)	44
26. CREATE INDEX (3 puan)	50

MS Excel'den MS SQL Server'a Tablo Aktarımı

MS SQL Server Management Studio 19.0.20209

Database, sağ tık, Tasks, Import Data...

Data source: Microsoft Excel

Excel file path (Browse...)

Excel version: Microsoft Excel 2016

Next

Destination: Microsoft OLE DB Provider for SQL Server

Next, Next, Next, Next, Finish

1. SELECT İfadesi, WHERE Yapısı, AND, OR, NOT Operatörleri (3 puan)

`SELECT * FROM table_name;`

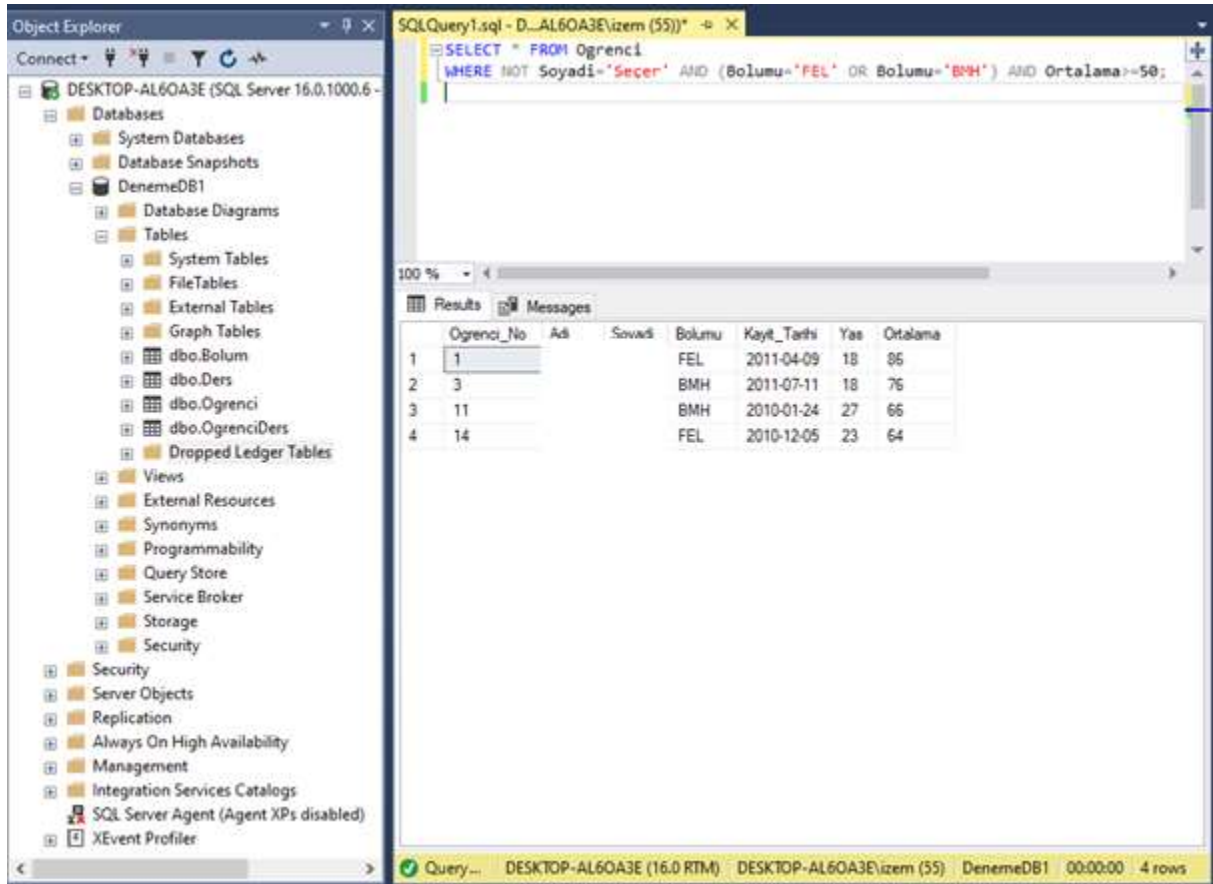
“SELECT” operatörünün “*” ve tablo adı ile birlikte kullanımı, veri tabanındaki ilgili tablonun tüm alanlarını seçer.

`SELECT column1, column2, ...
FROM table_name;`

Bir tablonun belirli kolonlarında yer alan kayıtların seçilmesi için “SELECT” operatörüne ilgili kolon isimleri parametre olarak gönderilir.

Aşağıdaki sorgu ile Öğrenci tablosunda yer alan öğrencilerden Seçer soyadında olmayan, ortalaması 50 ve üzerinde nota sahip ve BMH ya da FEL bölümü öğrencileri listelenmektedir.

`SELECT * FROM Öğrenci
WHERE NOT Soyadı='Seçer' AND (Bolumu='FEL' OR Bolumu='BMH') AND Ortalama>=50;`



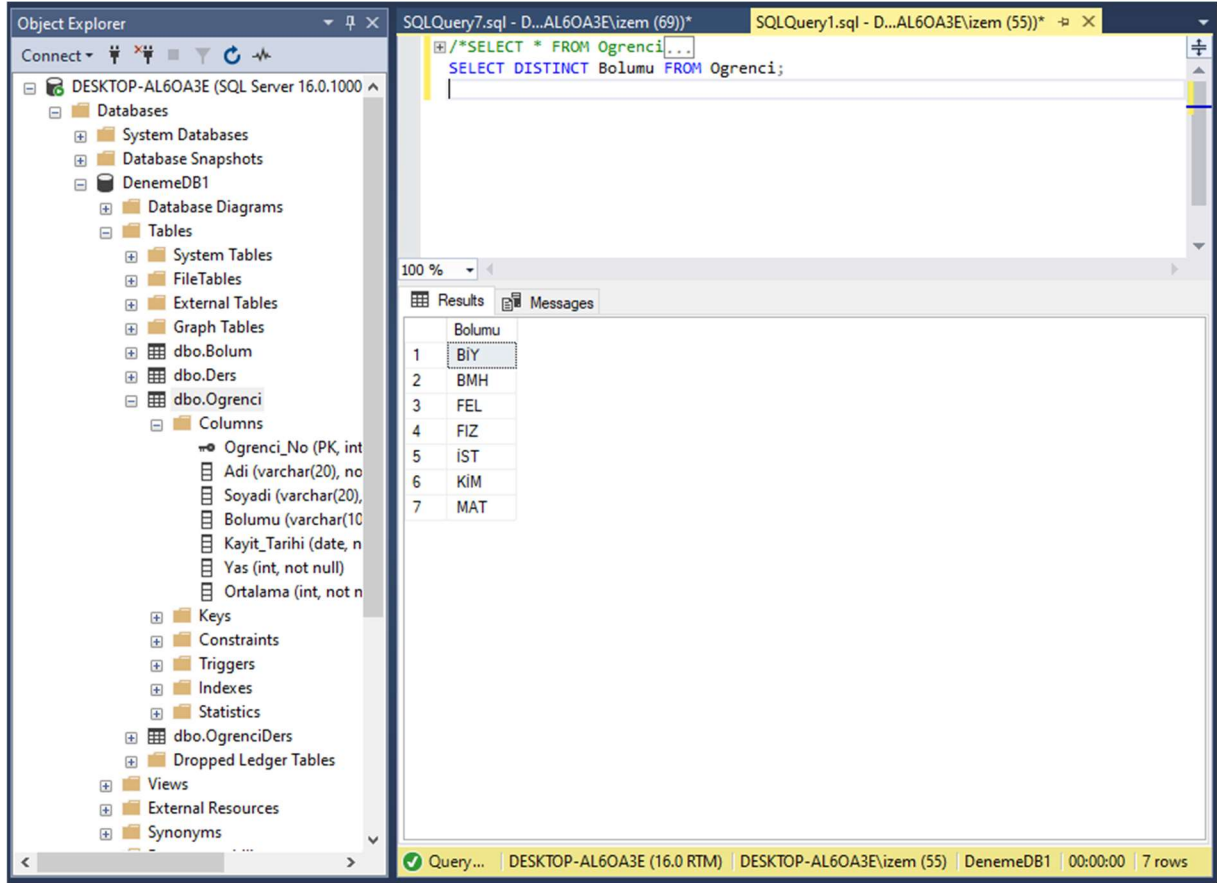
2. DISTINCT (3 puan)

*SELECT DISTINCT column1, column2, ...
FROM table_name;*

“SELECT” ifadesiyle birlikte kullanılan “DISTINCT” operatörü, kolon isimlerini parametre olarak alır. Bir kolonda tekrar eden kayıtları dışlayarak onların sadece farklı olan değerlerini döndürür.

Aşağıdaki sorgu ile Ogrenci tablosunda yer alan öğrencilerin bölümleri listelenmektedir.

SELECT DISTINCT Bolumu FROM Ogrenci;

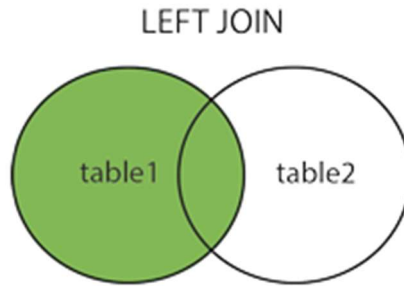


3. JOIN (LEFT JOIN, RIGHT JOIN, INNER JOIN) (7 puan)

“JOIN” ifadesi, birbiriyle ilgili olan iki veya daha fazla tablonun kolonlarının birleştirilmesi amacıyla kullanılır.

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

“SELECT” operatörüyle birlikte kullanılan “LEFT JOIN” ifadesi, en az iki tablo adı ile bunların birbiriyle eşleşen kolonlarını parametre olarak alır. “Table1” sol tabloyu, “Table2” ise sağ tabloyu ifade eder. “LEFT JOIN” sol tablodaki ilgili kolonun tüm kayıtlarını ve sağ tablodaki ilgili kolonun (sol tablonun ilgili kolonuyla) eşleşen kayıtlarını döndürür. Sağ tablonun ilgili kolonunda eşleşen kayıt yoksa, bunlar için “NULL” değerini döndürür.



Aşağıdaki sorgu ile OgrenciDers ve Ogrenci tabloları, Ogrenci_No kolonu üzerinden left join yöntemi ile birleştirilmektedir. Birleştirme işlemi OgrenciDers tablosunun ilgili tüm satırlarını

ve OgrenciDers ve Ogrenci tablolarının kesişim kümesini içermektedir. Birleştirilen yeni tabloda Ders_Kodu, (öğrenci) Adi ve Soyadı ile Sonuc kolonları yer almaktadır.

```
SELECT OgrenciDers.Ders_Kodu, Ogrenci.Adi, Ogrenci.Soyadi, OgrenciDers.Sonuc
FROM OgrenciDers
LEFT JOIN Ogrenci
ON OgrenciDers.Ogrenci_No=Ogrenci.Ogrenci_No;
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL60A3E (SQL Server 16.0.1000.6 - DenemeDB1)'. The 'Tables' folder is expanded, showing 'dbo.OgrenciDers' and 'dbo.Ogrenci'. The main window displays a SQL query in the 'SQLQuery7.sql' file:

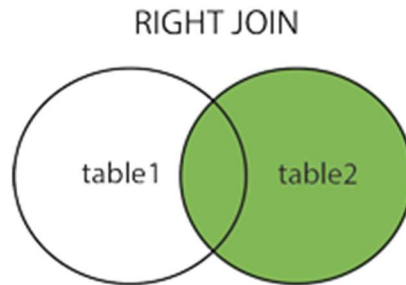
```
SELECT * FROM OgrenciDers
LEFT JOIN Ogrenci
ON OgrenciDers.Ogrenci_No=Ogrenci.Ogrenci_No;
```

The 'Results' pane shows the output of the query, which is a table with 30 rows and 5 columns: Ders_Kodu, Adi, Soyadi, Sonuc. The data is as follows:

Ders_Kodu	Adi	Soyadi	Sonuc
1	FIZ01		Kald
2	MUH01		Geçti
3	AYH01		Kald
4	ATA01		Kald
5	TBT01		Geçti
6	MAT02		Geçti
7	ATA01		Kald
8	TUD01		Geçti
9	TUD02		Kald
10	TUD01		Geçti
11	MUH01		Geçti
12	TBT01		Geçti
13	MAT01		Kald
14	TUD02		Geçti
15	MAT02		Geçti
16	TBT01		Geçti
17	MAT01		Kald
18	YAD01		Geçti
19	YAD02		Geçti
20	TBT01		Kald

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```

“SELECT” operatörüyle birlikte kullanılan “RIGHT JOIN” ifadesi, en az iki tablo adı ile bunların birbiriyle eşleşen kolonlarını parametre olarak alır. “Table1” sol tabloyu, “Table2” ise sağ tabloyu ifade eder. “RIGHT JOIN” sağ tablodaki ilgili kolonun tüm kayıtlarını ve sol tablodaki ilgili kolonun (sağ tablonun ilgili kolonuyla) eşleşen kayıtlarını döndürür. Sol tablonun ilgili kolonunda eşleşen kayıt yoksa, bunlar için “NULL” değerini döndürür.



Aşağıdaki sorgu ile OgrenciDers ve Ogrenci tabloları, Ogrenci_No kolonu üzerinden right join yöntemi ile birleştirilmektedir. Birleştirme işlemi Ogrenci tablosunun ilgili tüm satırlarını

ve OgrenciDers ve Ogrenci tablolarının kesişim kümesini içermektedir. Birleştirilen yeni tabloda Ders_Kodu, (öğrenci) Adi ve Soyadi ile Sonuc kolonları yer almaktadır.

```
SELECT OgrenciDers.Ders_Kodu, Ogrenci.Adi, Ogrenci.Soyadi, OgrenciDers.Sonuc
FROM OgrenciDers
RIGHT JOIN Ogrenci
ON OgrenciDers.Ogrenci_No=Ogrenci.Ogrenci_No;
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL60A3E (SQL Server 16.0.1000.6 - DenemeDB1)'. The 'Tables' folder is expanded, showing 'dbo.OgrenciDers' and 'dbo.Ogrenci'. The main window displays a SQL query in the 'SQLQuery7.sql' file:

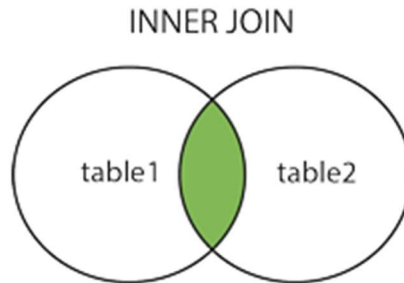
```
/*SELECT * FROM Ogrenci...
SELECT OgrenciDers.Ders_Kodu, Ogrenci.Adi, Ogrenci.Soyadi, OgrenciDers.Sonuc
FROM OgrenciDers
RIGHT JOIN Ogrenci
ON OgrenciDers.Ogrenci_No=Ogrenci.Ogrenci_No;
```

The 'Results' pane shows the output of the query, which is a table with 20 rows and 5 columns: Ders_Kodu, Adi, Soyadi, and Sonuc. The data is as follows:

Ders_Kodu	Adi	Soyadi	Sonuc
KIM01			Geçti
YAD02			Geçti
ATA01			Kaldı
MAT01			Kaldı
MAT01			Kaldı
MAT02			Geçti
TÜD01			Geçti
YAD01			Geçti
YAD02			Geçti
MUH01			Geçti
NULL			NULL
ATA01			Kaldı
TBT01			Geçti
HKG01			Geçti
TÜD02			Kaldı
TBT01			Kaldı
TÜD02			Geçti
YAD01			Geçti
MUH01			Geçti
CIZ01			Kaldı

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```

Benzer şekilde “SELECT” operatörüyle birlikte kullanılan “INNER JOIN” ifadesi, en az iki tablo adı ile bunların birbiriyle eşleşen kolonlarını parametre olarak alır. Her iki tablonun ilgili kolonlarda eşleşen değerlerini döndürür.



Aşağıdaki sorgu ile OgrenciDers ve Ogrenci tabloları, Ogrenci_No kolonu üzerinden inner join yöntemi ile birleştirilmektedir. Birleştirme işlemi OgrenciDers ve Ogrenci tablolarının kesişim kümesini içermektedir. Birleştirilen yeni tabloda Ogrenci_No, Ders_Kodu, Bolumu ve Sonuc kolonları yer almaktadır.

```
SELECT OgresciDers.Ogresci_No, OgresciDers.Ders_Kodu,
Ogresci.Bolumu, OgresciDers.Sonuc
FROM OgresciDers
INNER JOIN Ogresci
ON OgresciDers.Ogresci_No=Ogresci.Ogresci_No;
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DenemeDB1', including tables like 'dbo.Ogresci' and 'dbo.OgresciDers'. The main window shows a SQL query in the 'SQLQuery1.sql' file. The query is an inner join between 'OgresciDers' and 'Ogresci' tables, selecting columns 'Ogresci_No', 'Ders_Kodu', 'Bolumu', and 'Sonuc'. The results are displayed in a table with 20 rows and 4 columns. The first row is highlighted.

	Ogresci_No	Ders_Kodu	Bolumu	Sonuc
1	9	FIZ01	KIM	Kaldi
2	8	MUH01	KIM	Geçti
3	11	AYH01	BMH	Kaldi
4	5	ATA01	BMH	Kaldi
5	9	TBT01	KIM	Geçti
6	3	MAT02	BMH	Geçti
7	12	ATA01	FEL	Kaldi
8	15	TUD01	IST	Geçti
9	6	TUD02	MAT	Kaldi
10	3	TUD01	BMH	Geçti
11	9	MUH01	KIM	Geçti
12	12	TBT01	FEL	Geçti
13	2	MAT01	FIZ	Kaldi
14	7	TUD02	BIY	Geçti
15	13	MAT02	FIZ	Geçti
16	5	TBT01	BMH	Geçti
17	2	MAT01	FIZ	Kaldi
18	3	YAD01	BMH	Geçti
19	9	YAD02	KIM	Geçti
20	6	TRT01	MAT	Kaldi

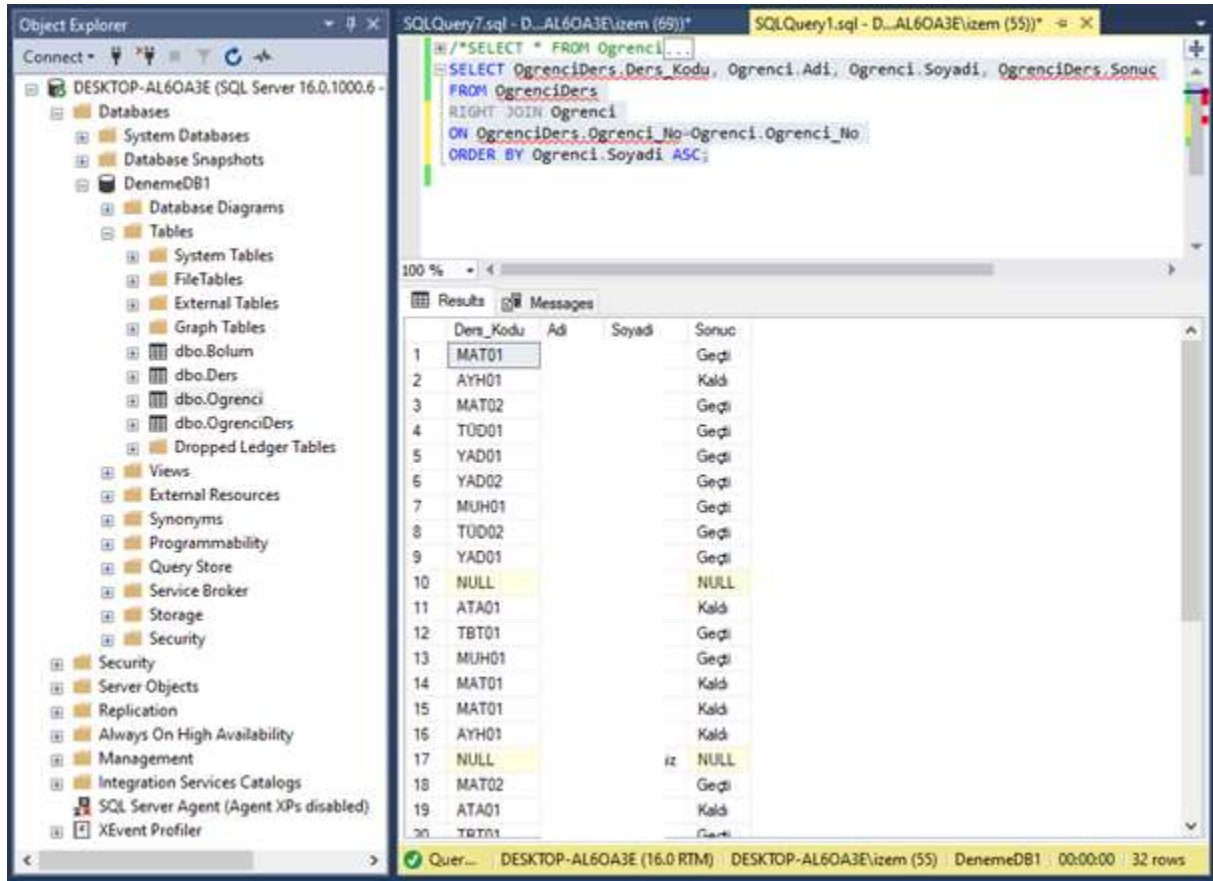
4. ORDER BY (3 puan)

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC/DESC;
```

“SELECT” operatörüyle birlikte kullanılan “ORDER BY” ifadesi, “ASC” ya da “DESC” ifadeleriyle azalan ya da artan şekilde, parametre olarak aldığı kolonların değerlerini sıralar. “ASC” ya da “DESC” ifadeleri kullanılmazsa varsayılan olarak “ASC” (artan) ifadesi geçerlidir.

Aşağıdaki sorguda right join ile elde edilen yeni tablo, öğrenci soy isimlerine göre A’ dan Z’ye (artan şekilde) sıralanmıştır.

```
SELECT OgresciDers.Ders_Kodu, Ogresci.Adı, Ogresci.Soyadı, OgresciDers.Sonuc
FROM OgresciDers
RIGHT JOIN Ogresci
ON OgresciDers.Ogresci_No=Ogresci.Ogresci_No
ORDER BY Ogresci.Soyadı ASC;
```



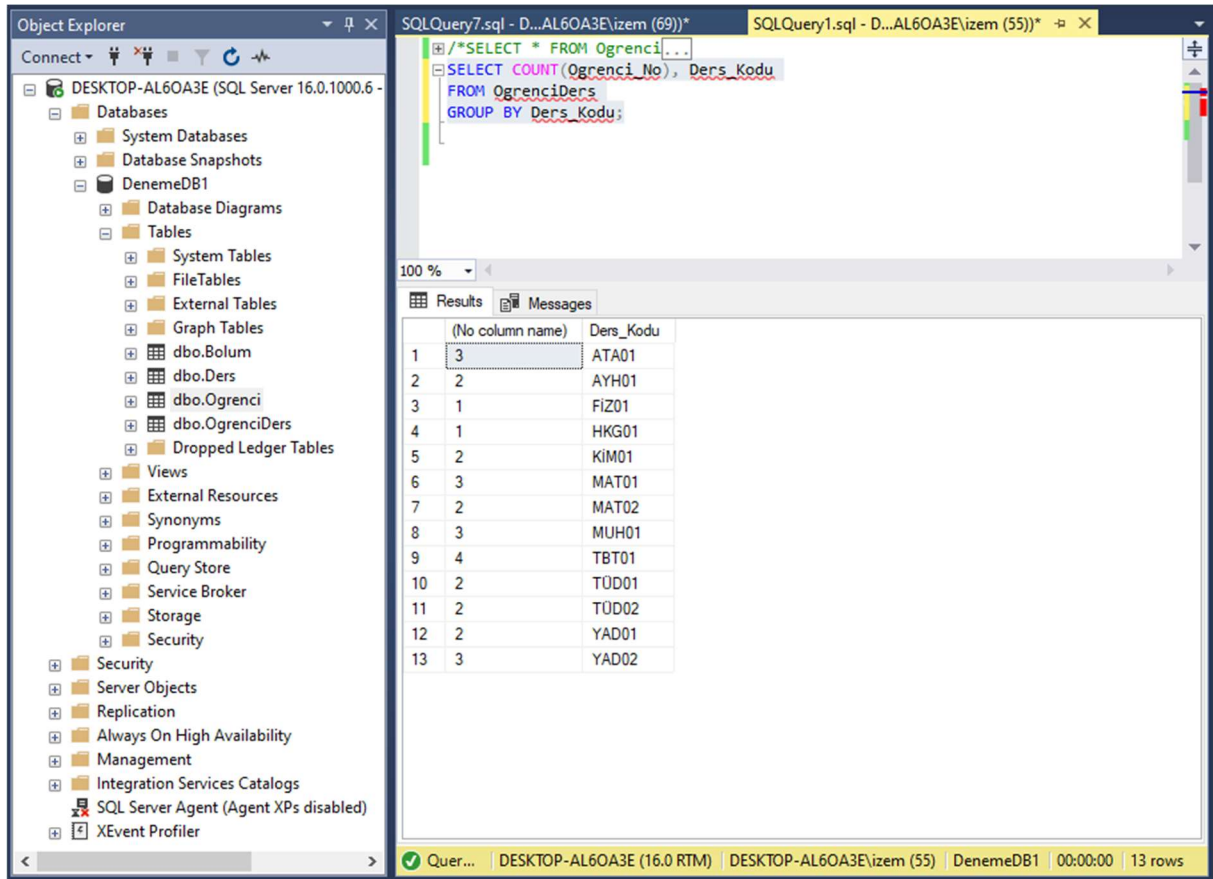
5. GROUP BY (3 puan)

```
SELECT column_name(s)
FROM table_name
GROUP BY column_name(s)
```

SELECT” operatörüyle birlikte kullanılan “GROUP BY” ifadesi, parametre olarak kolon ismi ya da isimleri alır. Özetlenen satırlarda aynı değerlere sahip satırları gruplandırma kullanılır. Genellikle COUNT(), MAX(), MIN(), SUM(), AVG() gibi toplulaştırma/özetleme ifadeleriyle birlikte kullanılır.

Aşağıdaki sorguda OgrenciDers tablosunda yer alan Ogrenci_No kayıtları sayılmış ve Ders_Kodu kayıtlarına göre gruplanmıştır. Böylece derslere kaydolun öğrenci sayıları, diğer bir ifadeyle hangi dersi kaç öğrencinin aldığı bilgisi elde edilmiştir.

```
SELECT COUNT(Ogrenci_No), Ders_Kodu
FROM OgrenciDers
GROUP BY Ders_Kodu;
```



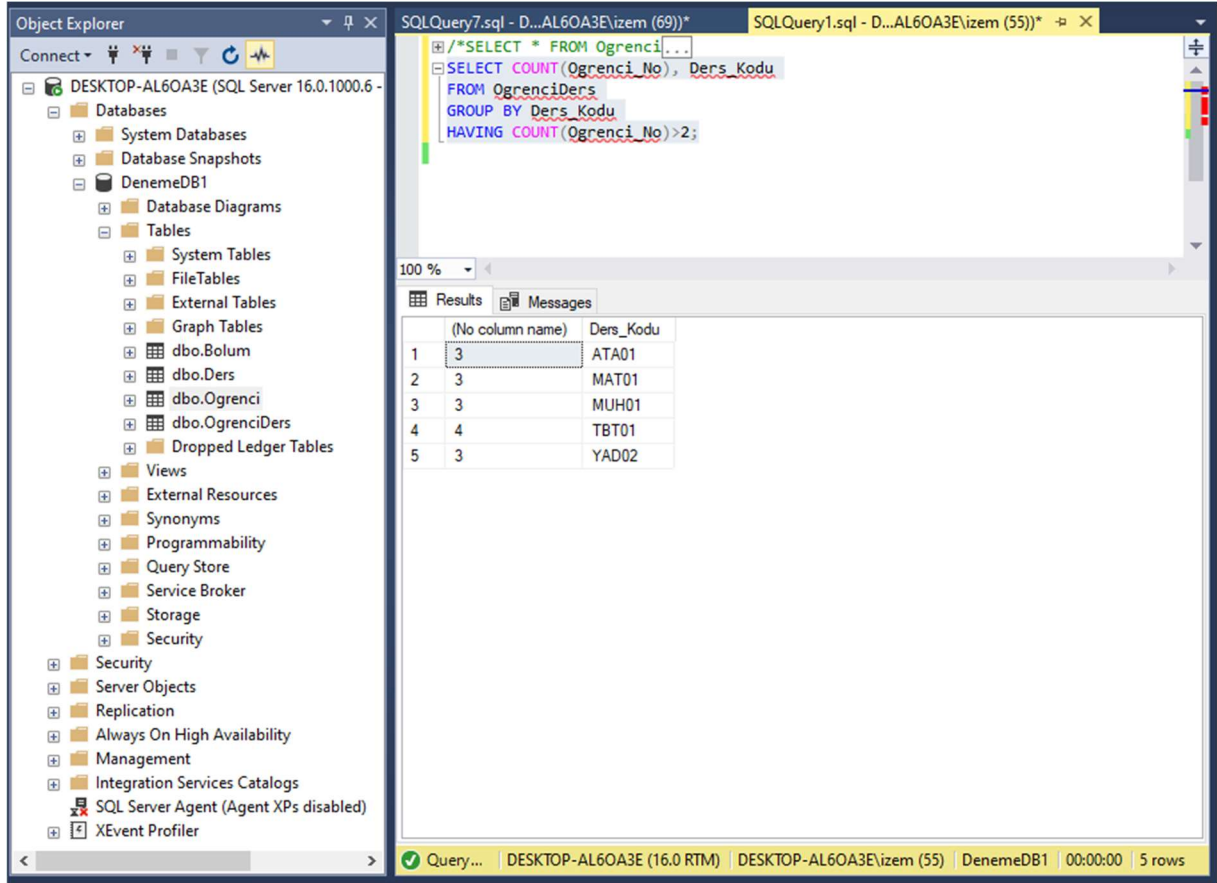
6. HAVING (3 puan)

*SELECT column_name(s)
FROM table_name
HAVING condition;*

SELECT” operatörüyle birlikte kullanılan “HAVING” ifadesi, bir koşulu parametre olarak alır. Görevi “WHERE” ifadesine benzemekle birlikte, “WHERE” ifadesi toplulaştırma/özetleme fonksiyonlarında kullanılamadığı için SQL diline eklenmiştir.

Aşağıdaki sorgu, bir önceki “GROUP BY” sorgusundan türetilmiş olup 2’den daha fazla öğrenci tarafından alınan dersleri listelemektedir.

*SELECT COUNT(Ogrenci_No), Ders_Kodu
FROM OgrenciDers
GROUP BY Ders_Kodu
HAVING COUNT(Ogrenci_No)>2;*



7. IN ve LIKE (3 puan)

*SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);*

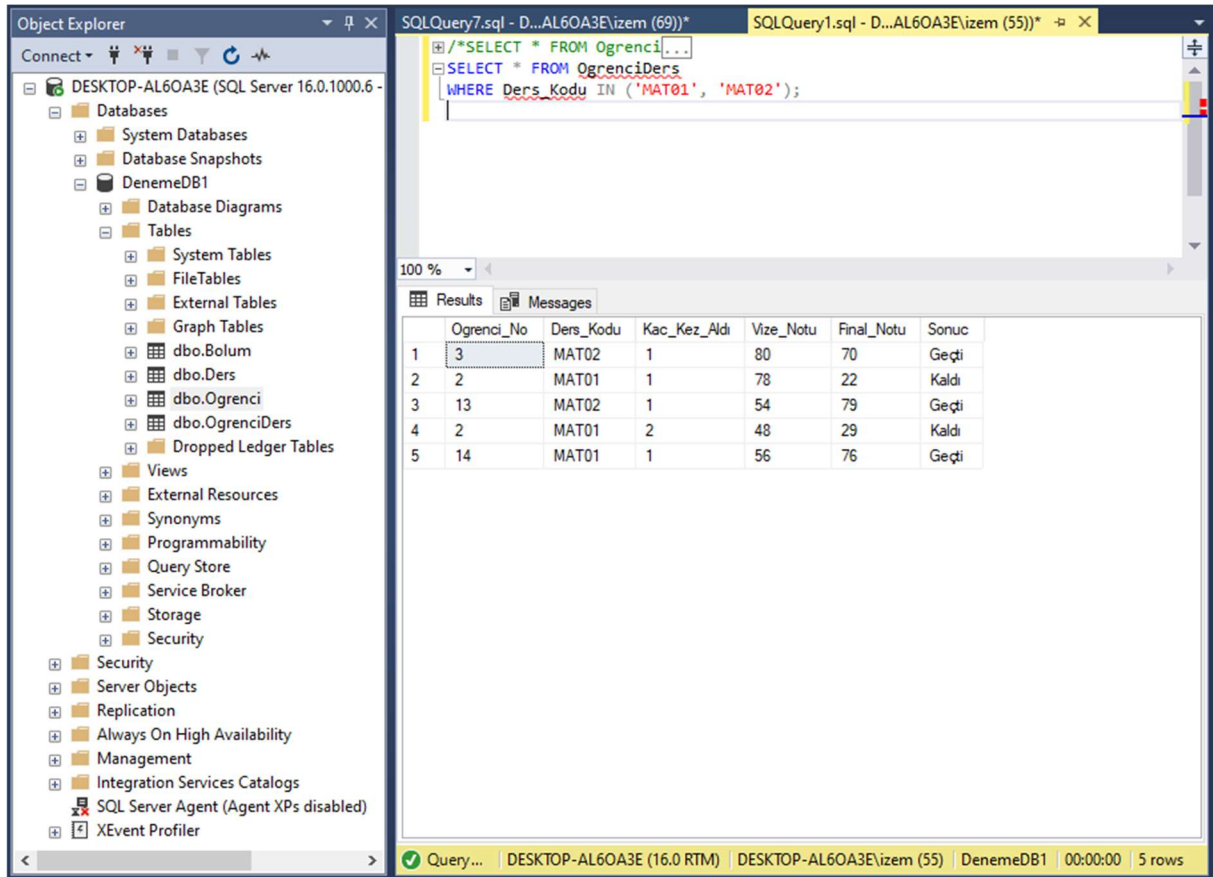
“SELECT” ve “WHERE” operatörleriyle birlikte kullanılan “IN” ifadesi bir kolonda yer alan değerleri parametre olarak alabilir. “IN” ifadesi, bir “WHERE” operatörü içinde çok sayıda değeri belirlemeye olanak tanır. “IN” ifadesi çoklu bir “OR” kullanımının kısaltılmasına karşılık gelir.

*SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);*

Yukarıda gösterildiği gibi “IN” ifadesi, bir kolonda yer alan değerler yerine bir “SELECT” ifadesini de parametre olarak alabilir.

Aşağıdaki sorguda, OgrenciDers tablosundan sadece MAT01 ya da MAT02 dersini alan öğrenciler filtrelenmiştir.

*SELECT * FROM OgrenciDers
WHERE Ders_Kodu IN ('MAT01', 'MAT02');*

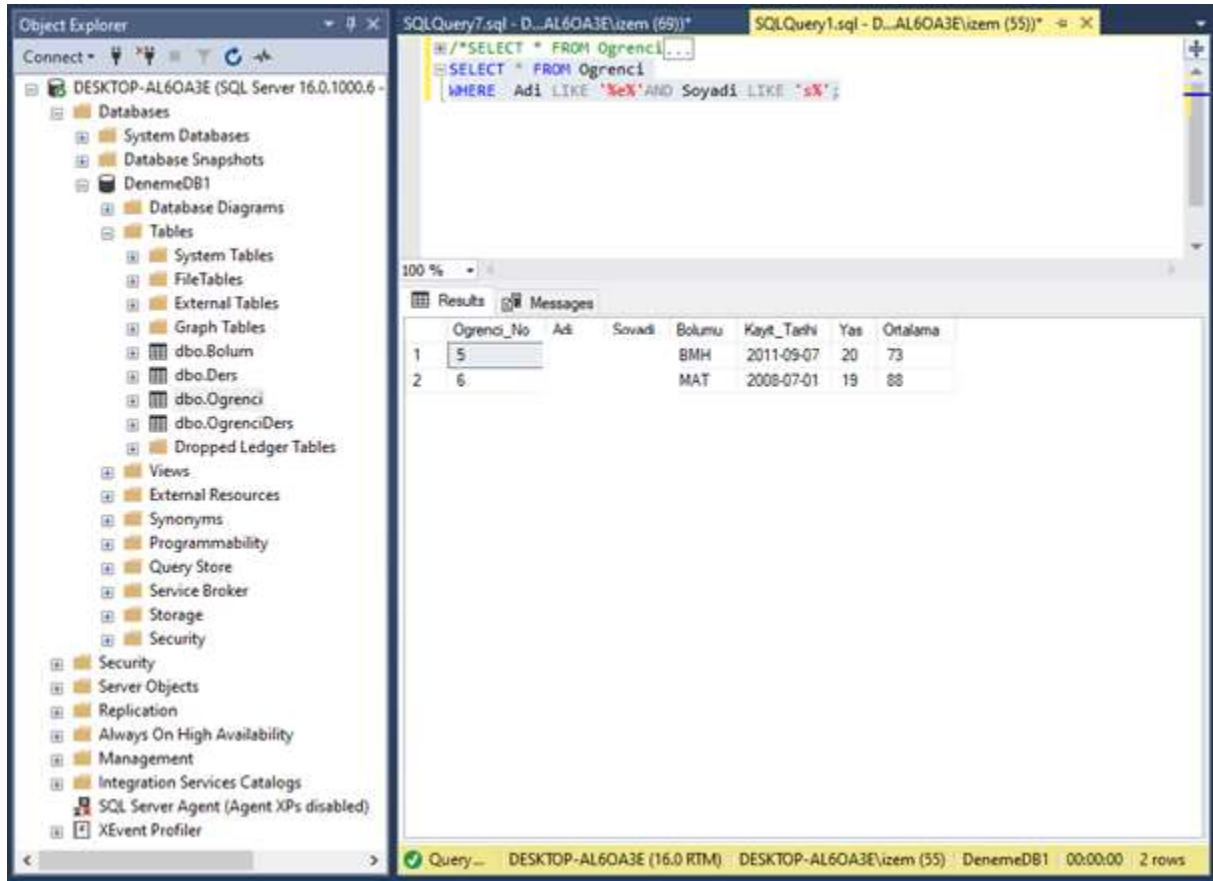


*SELECT column1, column2, ...
FROM table_name
WHERE column LIKE pattern;*

SELECT” ve “WHERE” operatörleriyle birlikte kullanılan “LIKE” ifadesi, “patern” kısmında “%” ya da “_” operatörleri veya “AND” ya da “OR” operatörleriyle kombini yer alabilir. Bir kolonda yer alan kayıtları belirli örüntülere göre filtrelemeye olanak tanır. Örneğin LIKE 'a%' “a” karakteri ile başlayan kayıtları, LIKE '%a' “a” karakteri ile biten kayıtları, LIKE '%or%' ise içinde “or” karakterleri geçen kayıtları döndürür. Benzer şekilde LIKE '_r%' ikinci karakteri “r” olan kayıtları ya da LIKE 'a__%' ise “a” karakteri ile başlayan en az 3 karakter içeren kayıtları döndürür.

Aşağıdaki sorguda, Ogrenci tablosunda yer alan kayıtlardan, isminde “e” karakteri bulunan ve soy ismi “s” karakteriyle başlayan öğrenci kayıtları filtrelendi.

*SELECT * FROM Ogrenci
WHERE Adi LIKE '%e%' AND Soyadi LIKE 's%';*



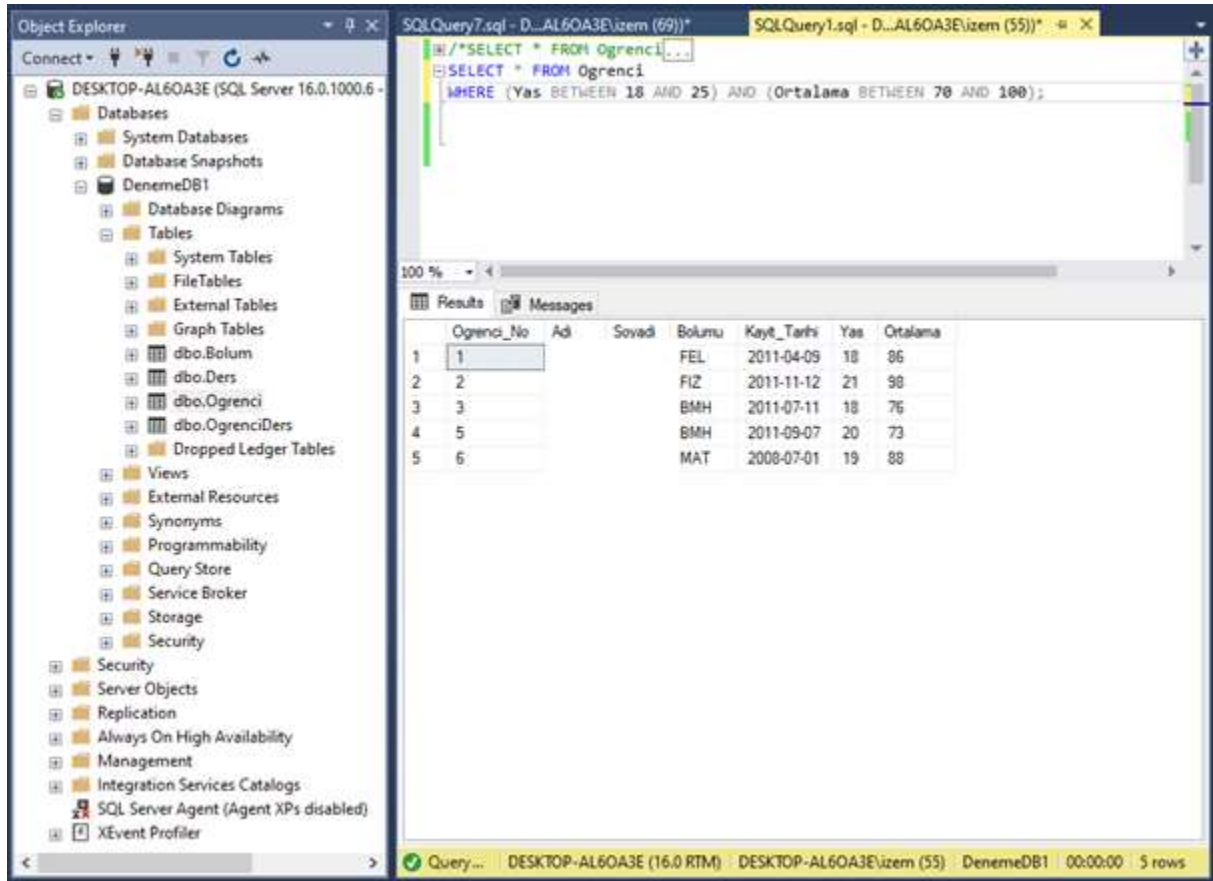
8. BETWEEN (3 puan)

*SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;*

SELECT”, WHERE ve “AND” operatörleriyle birlikte kullanılan “BETWEEN” ifadesi, başlangıç ve bitiş şeklinde iki değeri parametre olarak alır. İşleme başlangıç ve bitiş değerleri dahildir. Sayı, metin ya da tarih şeklinde olabilen değerleri, belirli bir aralıkta sınırlandırır. Opsiyonel olarak “NOT” ve “IN” operatörleriyle yaygın bir şekilde kullanılır.

Aşağıdaki sorguda 18-25 yaş aralığında olup da 70 ile 100 arasında ortalamaya sahip öğrenciler listelenmiştir.

*SELECT * FROM Ogrenci
WHERE (Yas BETWEEN 18 AND 25) AND (Ortalama BETWEEN 70 AND 100);*



9. UNION (3 puan)

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

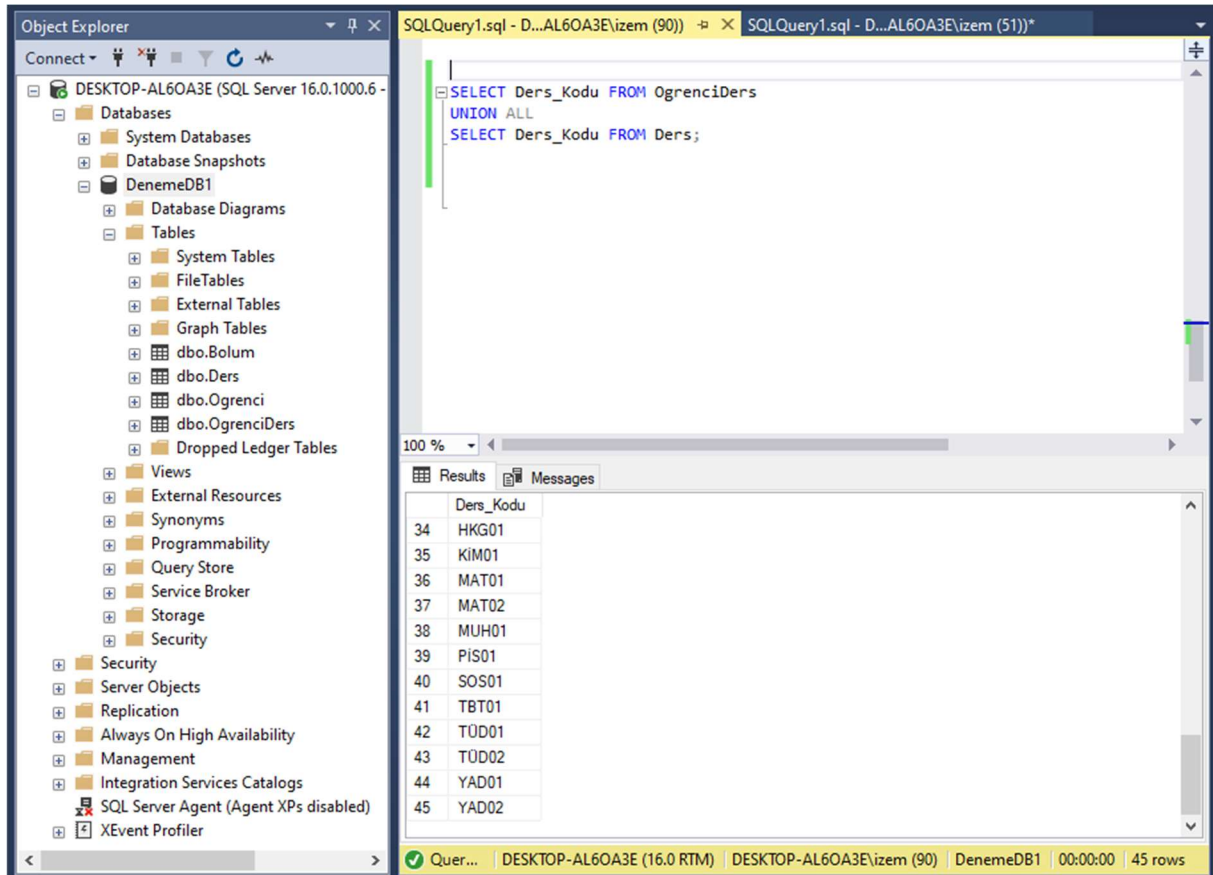
“SELECT” ile seçilen birden fazla tablonun benzer kolonlarında yer alan kayıtları tekrarsız bir şekilde birleştirir. Tabloları ve kolonları parametre olarak alır.

```
SELECT column_name(s) FROM table1
UNION ALL
SELECT column_name(s) FROM table2;
```

Yukarıdaki sentaksta “UNION ALL” ifadesi ise birden fazla tablonun benzer kolonlarında yer alan kayıtları tekrarlı bir şekilde birleştirir.

Aşağıdaki sorguda OgrenciDers tablosunda yer alan Ders_Kodu kolonu ile Ders tablosunda bulunan Ders_Kodu kolonu UNION ALL yöntemi ile birleştirilmiştir.

```
SELECT Ders_Kodu FROM OgrenciDers
UNION ALL
SELECT Ders_Kodu FROM Ders;
```



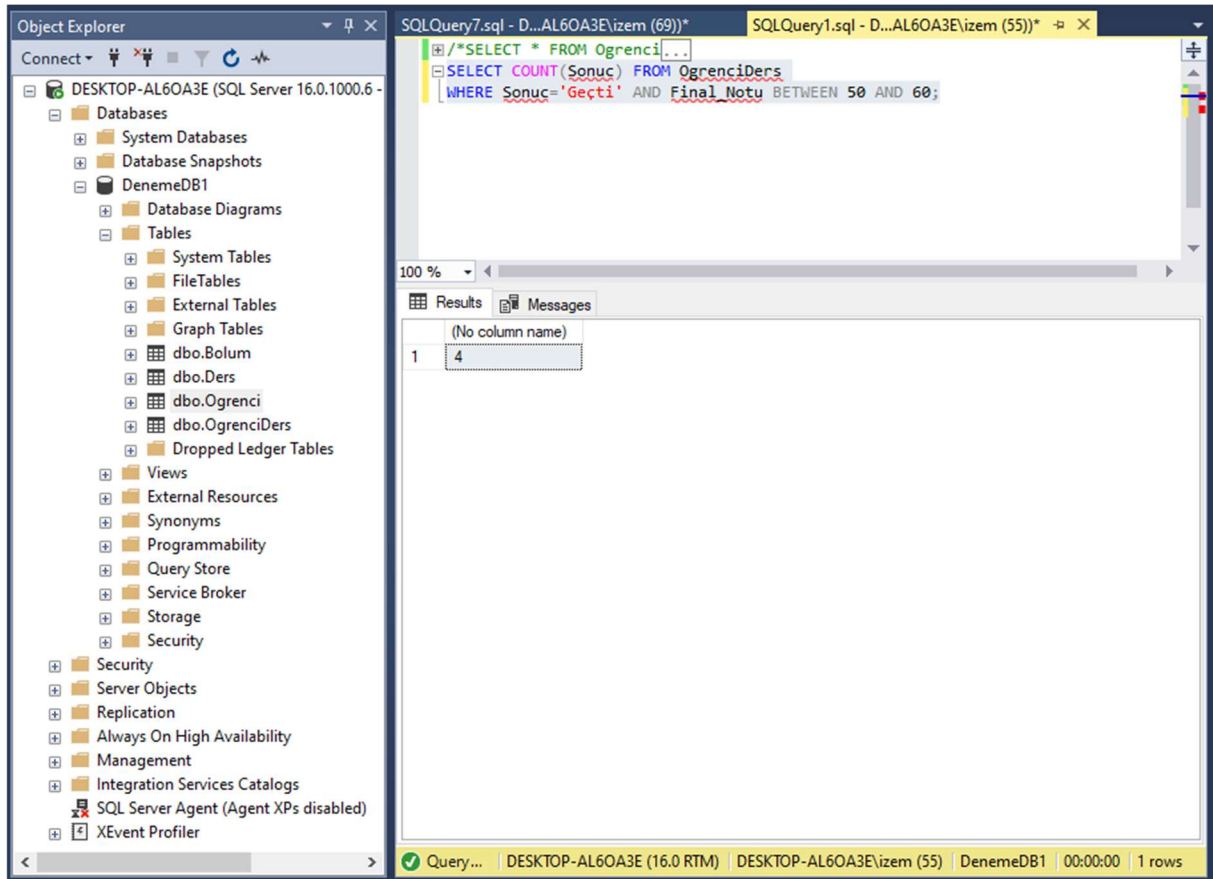
10. COUNT, SUM Fonksiyonları (3 puan)

*SELECT COUNT(column_name)
FROM table_name
WHERE condition;*

“COUNT” fonksiyonu parametre olarak bir kolon ismi alır ve (“WHERE” ifadesi gibi bir kriterle kombinlenerek) satır/kayıt sayısını döndürür.

OgrenciDers tablosu kullanılarak Final_Notu 50 ile 60 arasında olup da aldığı dersten geçen öğrenci sayısı aşağıda gösterilmiştir.

*SELECT COUNT(Sonuc) FROM OgrenciDers
WHERE Sonuc='Geçti' AND Final_Notu BETWEEN 50 AND 60;*

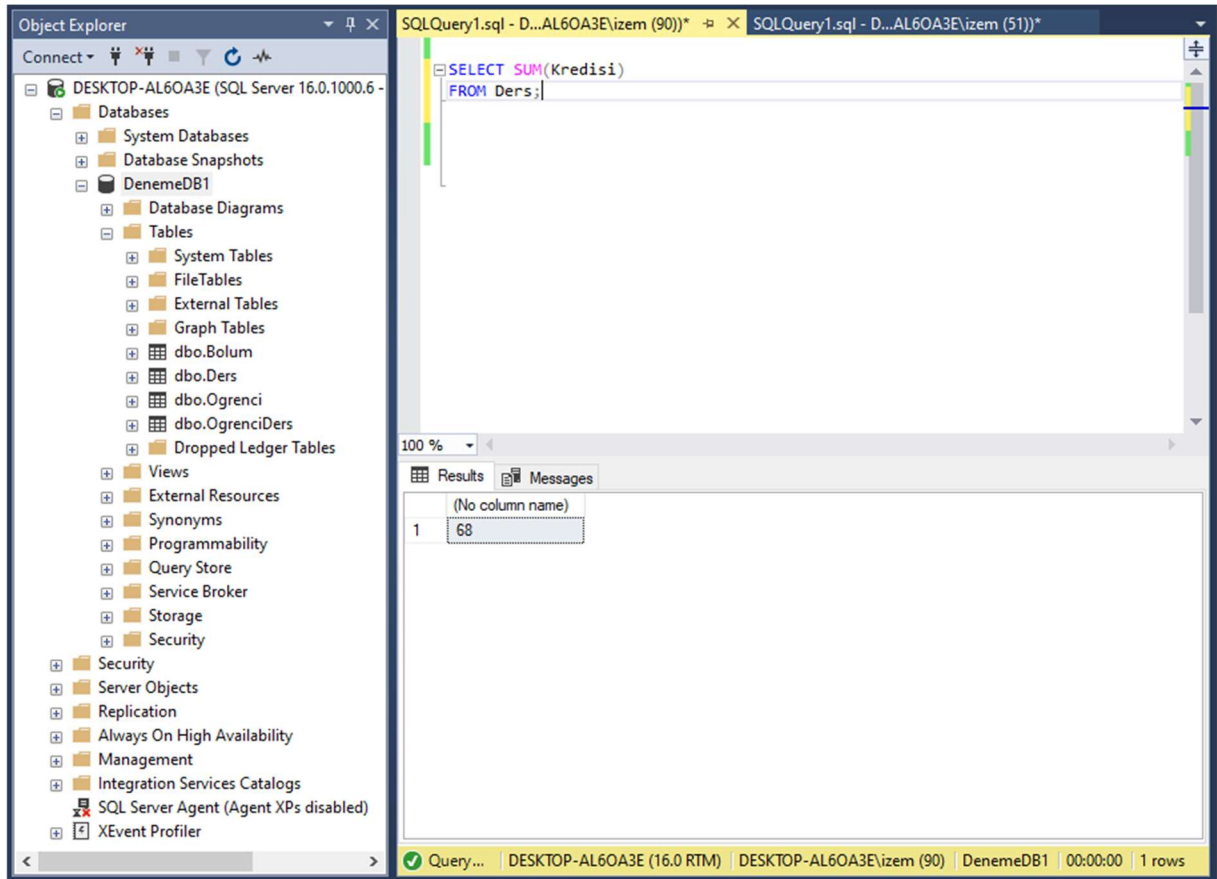


```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

“SUM” fonksiyonu da “COUNT” gibi parametre olarak bir kolon ismi alır ve “(“WHERE” ifadesi gibi bir kriterle kombinlenerek) numerik değerlerden oluşan kayıtların toplamını döndürür.

Aşağıdaki sorguda Ders tablosunda yer alan kredilerin toplamı gösterilmiştir.

```
SELECT SUM(Kredisi)
FROM Ders;
```



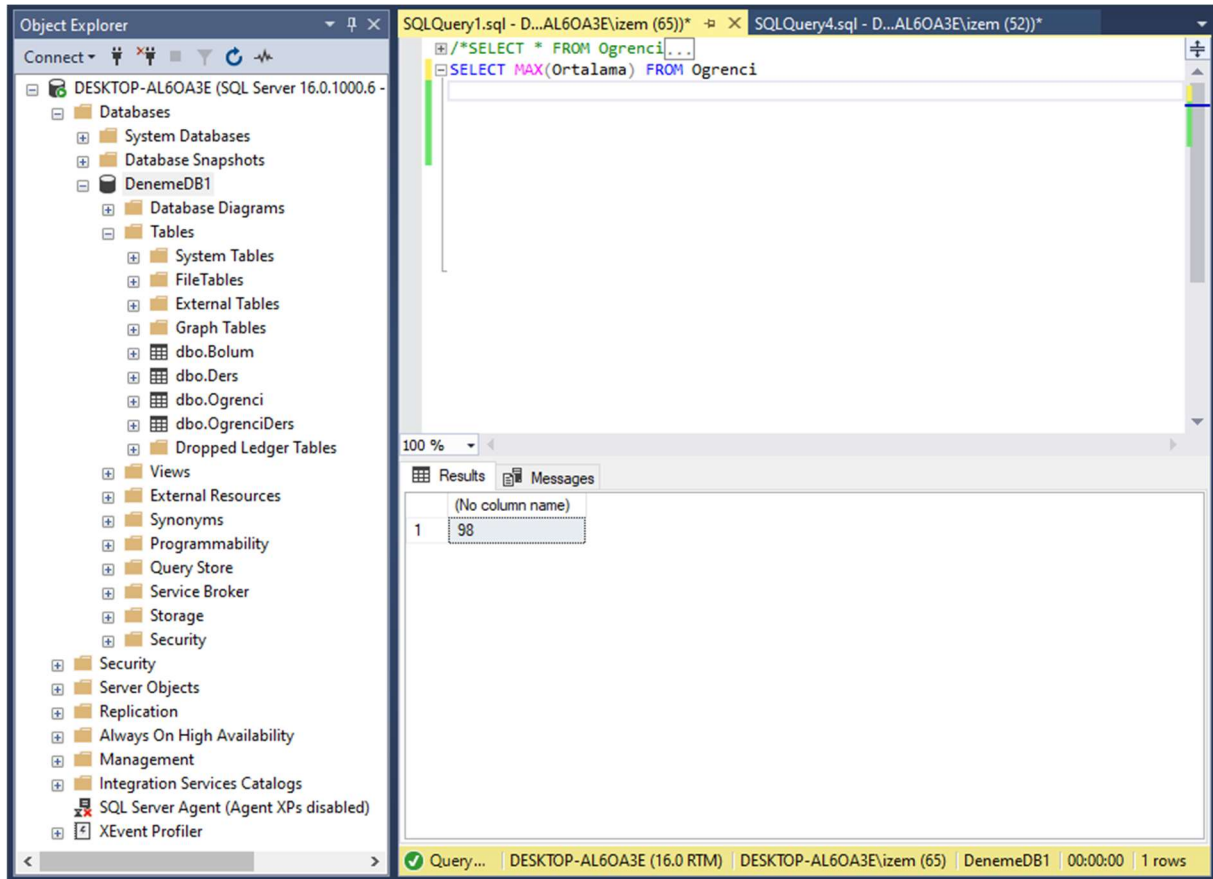
11. MAX, MIN, AVG Fonksiyonları (3 puan)

*SELECT MAX(column_name)
FROM table_name;*

“MAX” fonksiyonu parametre olarak kolon ismini alır. Numerik değerlerden oluşan seçilen kolonda yer alan kayıtların en büyüğünü döndürür.

Aşağıdaki sorguyla en yüksek ortalama not bilgisi Öğrenci tablosundan elde edilmiştir.

SELECT MAX(Ortalama) FROM Öğrenci

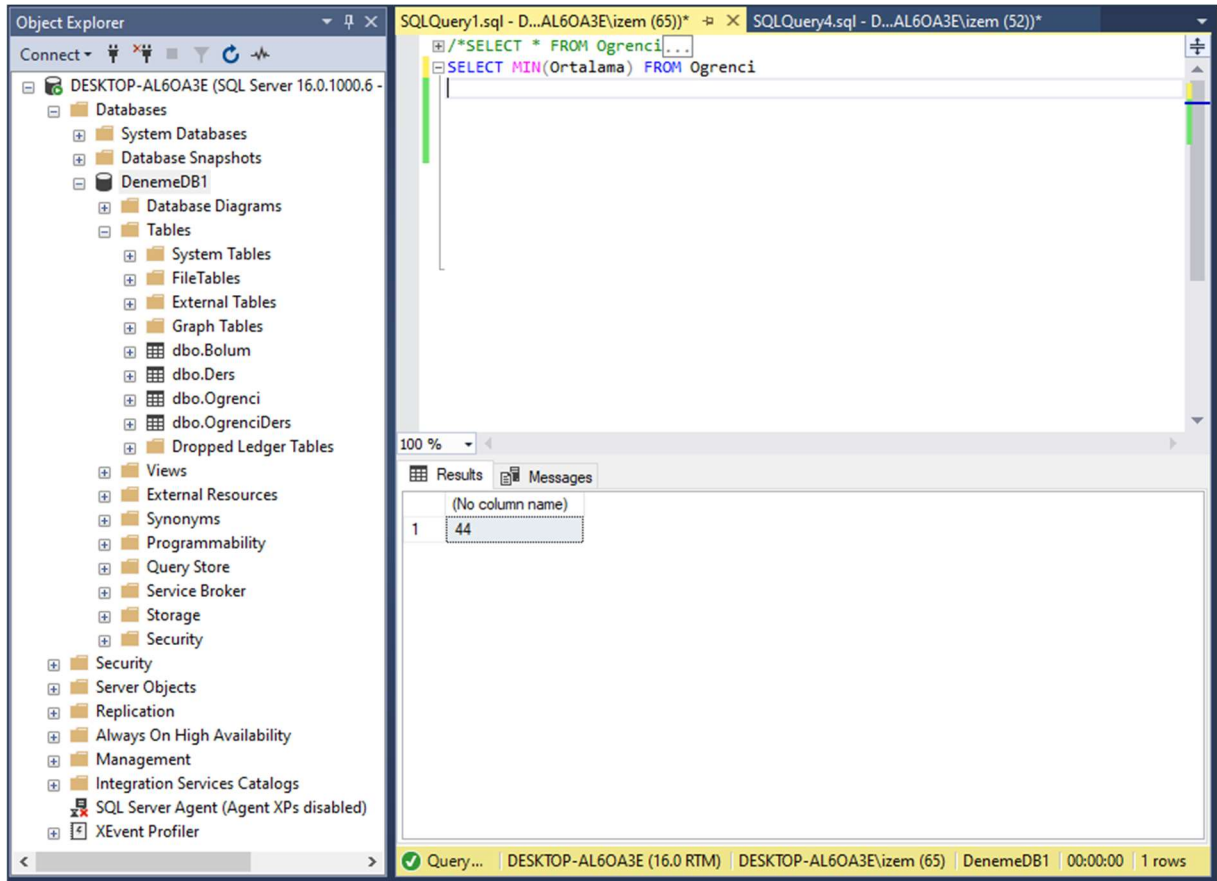


`SELECT MIN(column_name)
FROM table_name;`

“MIN” fonksiyonu parametre olarak kolon ismini alır. Numerik değerlerden oluşan seçilen kolonda yer alan kayıtların en küçüğünü döndürür.

Aşağıdaki sorguyla en düşük ortalama not bilgisi Ogrenci tablosundan elde edilmiştir.

`SELECT MIN(Ortalama) FROM Ogrenci`

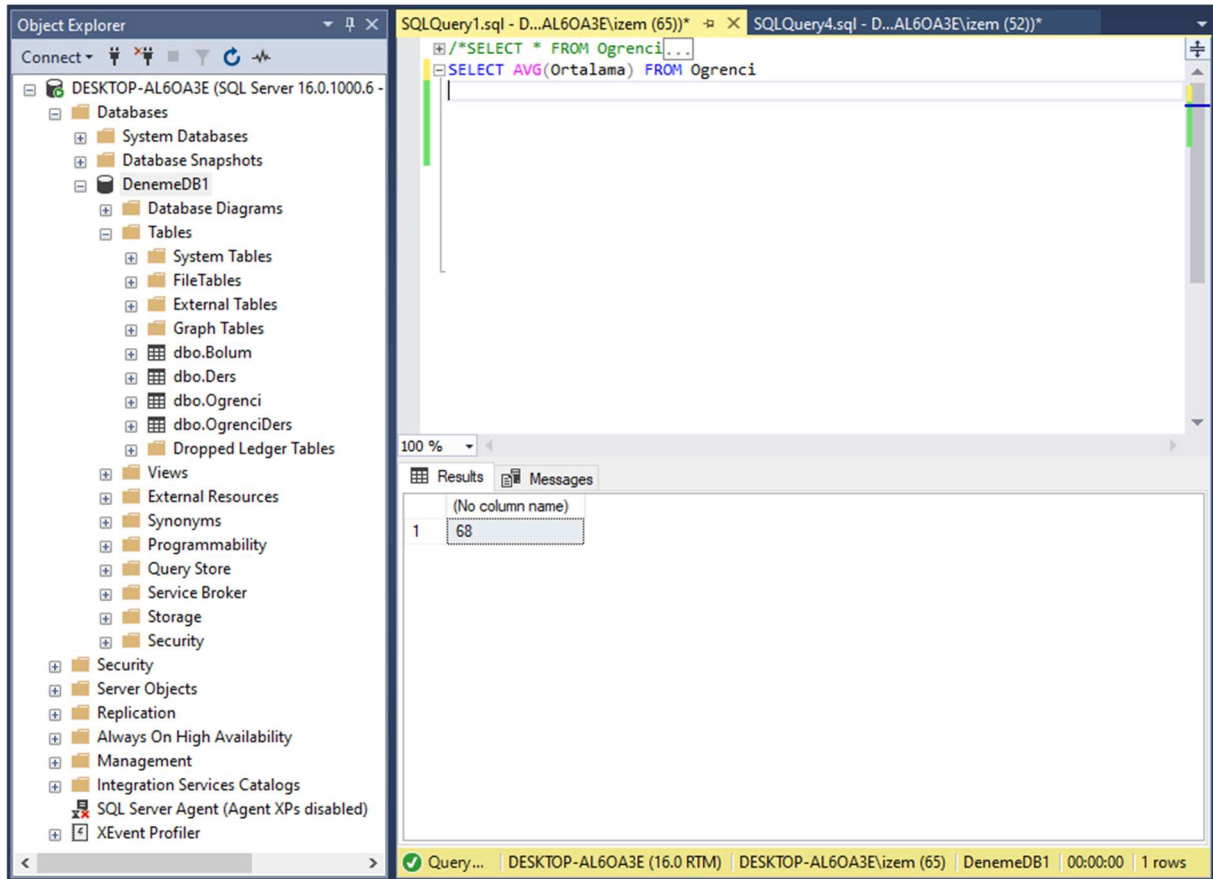


```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

“AVG” ifadesi parametre olarak bir kolon ismi alır ve “WHERE” ifadesi gibi bir kritere göre “numerik” değerlerden oluşan kayıtların ortalamasını döndürür.

Aşağıdaki sorguyla, Ogrenci tablosunun Ortalama kolonunda yer alan kayıtların ortalaması elde edilmiştir.

```
SELECT AVG(Ortalama) FROM Ogrenci
```



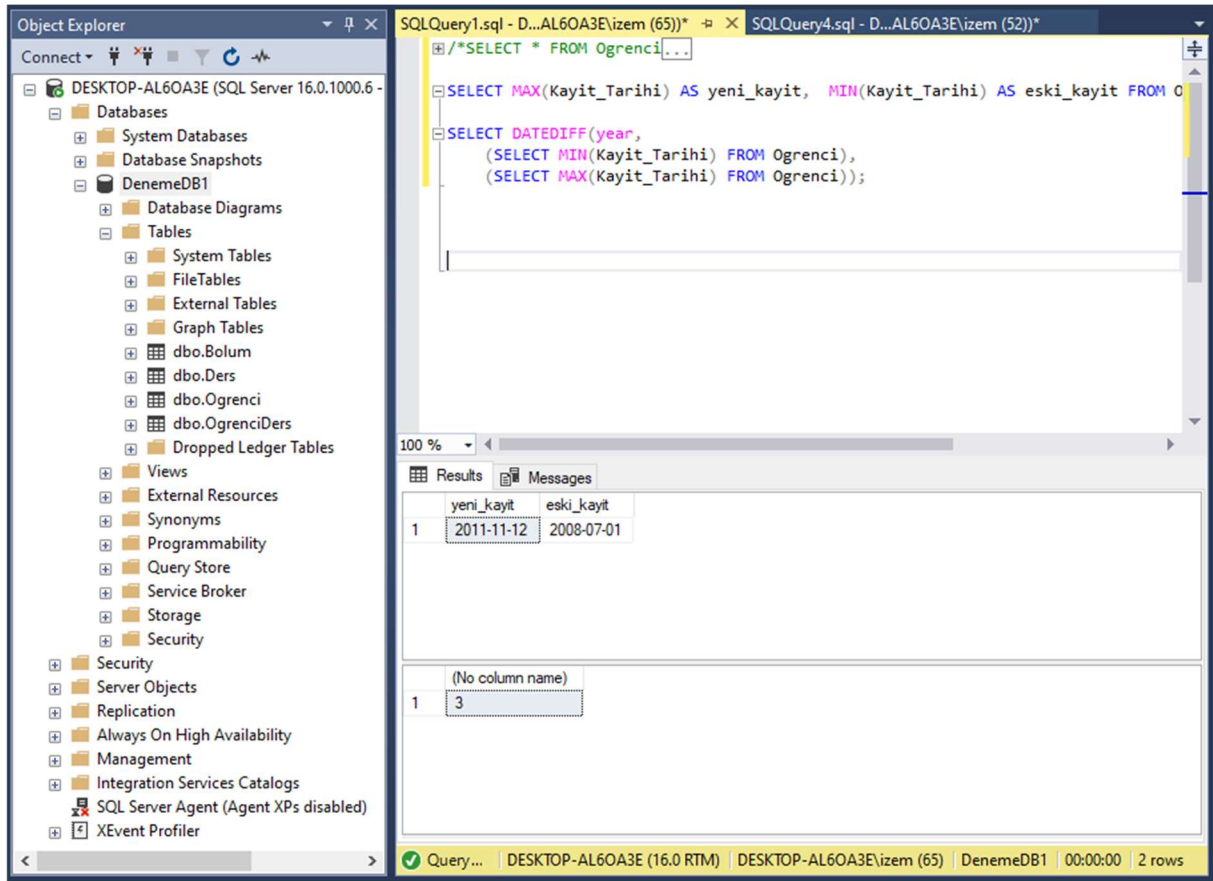
12. DATEDIFF ve DATEADD Fonksiyonları (3 puan)

DATEDIFF(interval, date1, date2)

“DATEDIFF” fonksiyonu, parametre olarak aldığı iki tarih arasındaki farkı, aldığı “interval” parametresiyle gün, ay, yıl gibi bir formatta döndürür.

Aşağıdaki sorguda önce en yeni ve eski kayıt tarihleri gösterilmiş, sonra da her iki tarih arasında yıl bazında fark alınmıştır.

```
SELECT MAX(Kayit_Tarihi) AS yeni_kayit, MIN(Kayit_Tarihi) AS eski_kayit FROM Ogrenci;  
SELECT DATEDIFF(year,  
    (SELECT MIN(Kayit_Tarihi) FROM Ogrenci),  
    (SELECT MAX(Kayit_Tarihi) FROM Ogrenci));
```



DATEADD(interval, number, date)

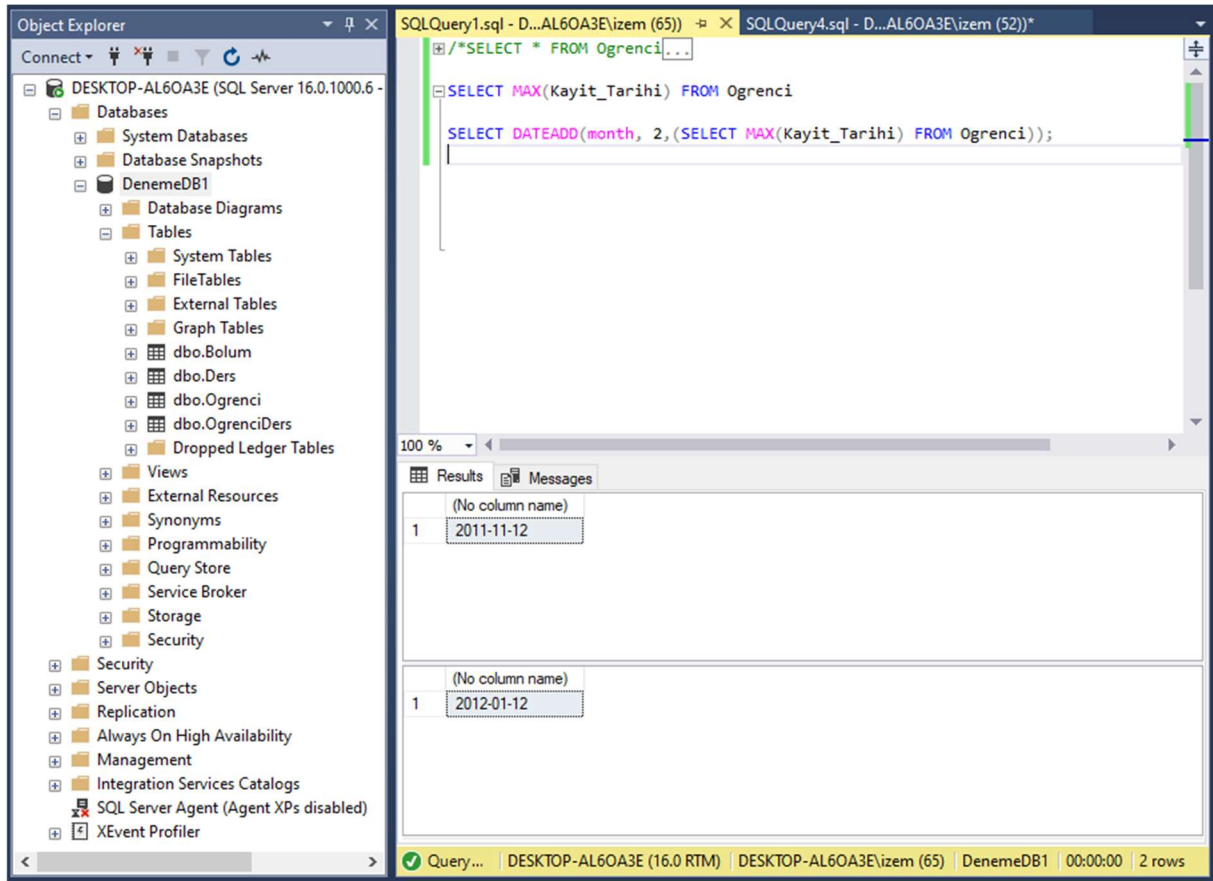
Benzer şekilde “DATEADD” fonksiyonu, tarih, sayı ve (gün, ay, yıl gibi) bir aralık/dönem (interval) değerini parametre olarak alır. Böylece bir tarihe, numerik bir değerden yararlanılarak dakika, saat, gün, hafta, ay gibi zaman/tarih değerleri ilave edilebilir.

Aşağıdaki sorguda önce en yeni kayıt tarihleri gösterilmiş, sonra da bu tarihe 2 ay eklenerek yeni bir tarih elde edilmiştir.

```

SELECT MAX(Kayit_Tarihi) FROM Ogrrenci;
SELECT DATEADD(month, 2, (SELECT MAX(Kayit_Tarihi) FROM Ogrrenci));

```



13. İç İç SELECT İfadesi (Nested Query) (6 puan)

```
SELECT *
FROM table_name
WHERE column_names > (
    SELECT AVG(column_names)
    FROM table_name);
```

İç içe “SELECT” kullanımına ilişkin yukarıdaki sentaks, tablo ve kolon adını parametre olarak alır. İç kısımda yer alan gömülü “SELECT” ifadesi, bir özetleme fonksiyonu olan AVG ile birlikte kullanılarak bir kolondaki numerik kayıtların ortalamasını döndürür. Dıştaki “SELECT” ifadesi ise gömülü “SELECT” ifadesinin döndürdüğü değeri, bir karşılaştırma operatörü yardımıyla ilgili kolonda yer alan kayıtların tümüyle karşılaştırarak nihai değeri ya da değerleri döndürür. İç içe “SELECT” ifadeleri ikiden daha fazla olabilir. Benzer şekilde birden fazla tablo ya da kolon adını parametre olarak alabilir.

Aşağıdaki iç içe SELECT sorgusunda final notu, ortalama notların ortalamasından büyük olan öğrenciler listelenmektedir. Bu sonuca göre final notu, vize ve final ağırlıklarını temsil eden ortalama notların aritmetik ortalamasından büyük olan öğrencilerin genel olarak başarılı olduğu söylenebilir.

```
SELECT * FROM OgrenciDers
WHERE Final_Notu > (
    SELECT AVG(Ortalama)
    FROM Ogrenci);
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL60A3E (SQL Server 16.0.1000.6 - DenemeDB1)'. The 'Tables' folder is expanded, showing 'dbo.OgrenciDers'. The main window displays the following SQL query:

```
SELECT * FROM OgrenciDers
WHERE Final_Notu > (
    SELECT AVG(Ortalama)
    FROM Ogrenci
);
```

Below the query window, the 'Results' tab shows the output of the query. The results are displayed in a table with the following columns: 'Ogrenci_No', 'Ders_Kodu', 'Kac_Kez_Aldi', 'Vize_Notu', 'Final_Notu', and 'Sonuc'. The table contains 13 rows of data.

Ogrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc
2	15	TUD01	1	25	84
3	9	MUH01	1	58	80
4	12	TBT01	1	93	80
5	7	TUD02	1	43	99
6	13	MAT02	1	54	79
7	5	TBT01	1	55	96
8	3	YAD01	1	55	97
9	14	MAT01	1	56	76
10	3	MUH01	1	47	83
11	1	YAD02	1	81	85
12	5	HKG01	1	85	74
13	7	YAD01	1	43	72

The status bar at the bottom indicates that the query was executed successfully, returning 13 rows.

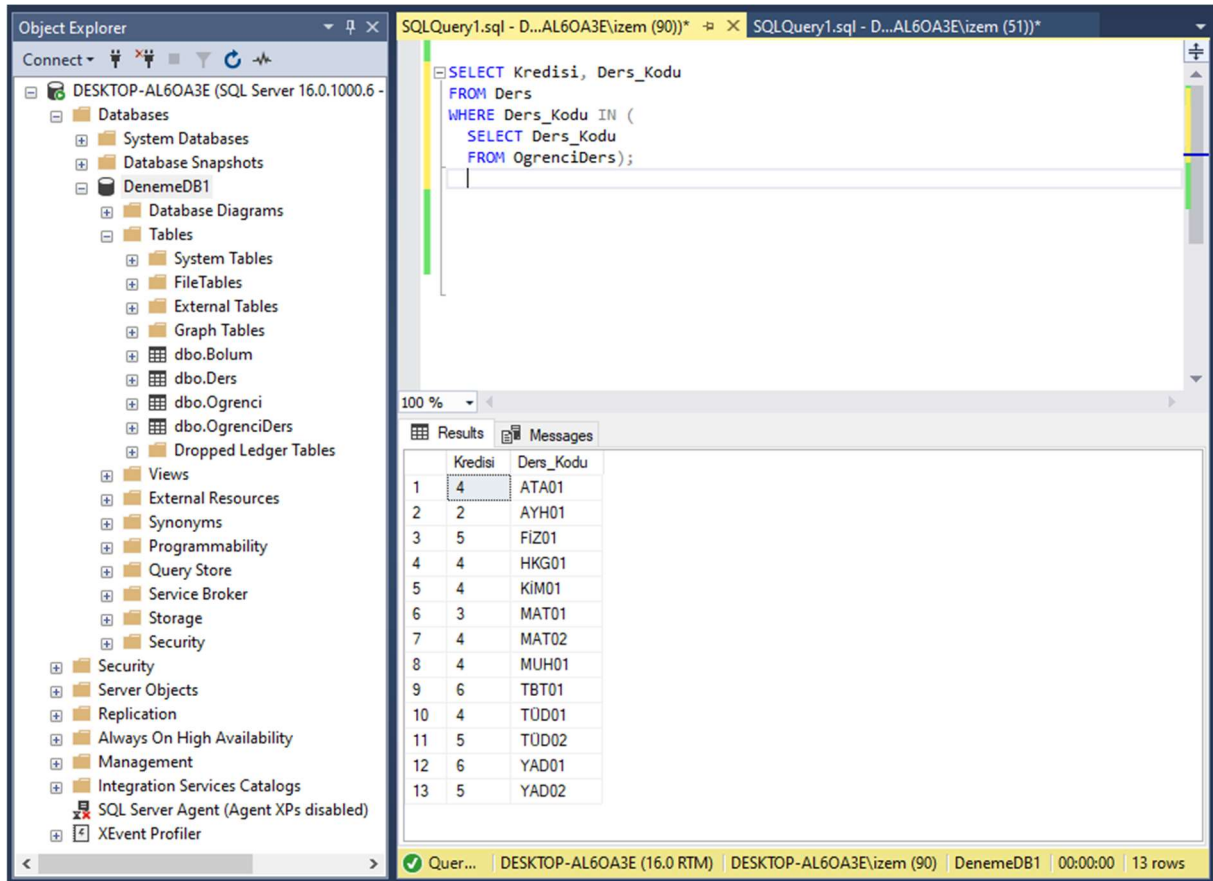
Aşağıdaki sentaksta ise içi içe “SELECT” ifadesinin “IN” ile kullanımı örneklenmiştir.

```
SELECT column_name1, column_name2, ...
FROM table_name1
WHERE column_name1 IN (
    SELECT column_name1
    FROM table_name2);
```

Yukarıdaki sentaks iki farklı tablo adını ve birden fazla kolon adını parametre olarak almaktadır.

Aşağıdaki sorguyla Ders tablosunda yer alan derslerden hangilerinin öğrenciler tarafından alındığı (OgrenciDers tablosunda yer aldığı) tespit edilmektedir. Diğer taraftan, öğrenciler tarafından alınan derslerin kredileri de listelenmektedir.

```
SELECT Kredisi, Ders_Kodu
FROM Ders
WHERE Ders_Kodu IN (
    SELECT Ders_Kodu
    FROM OgrenciDers);
```



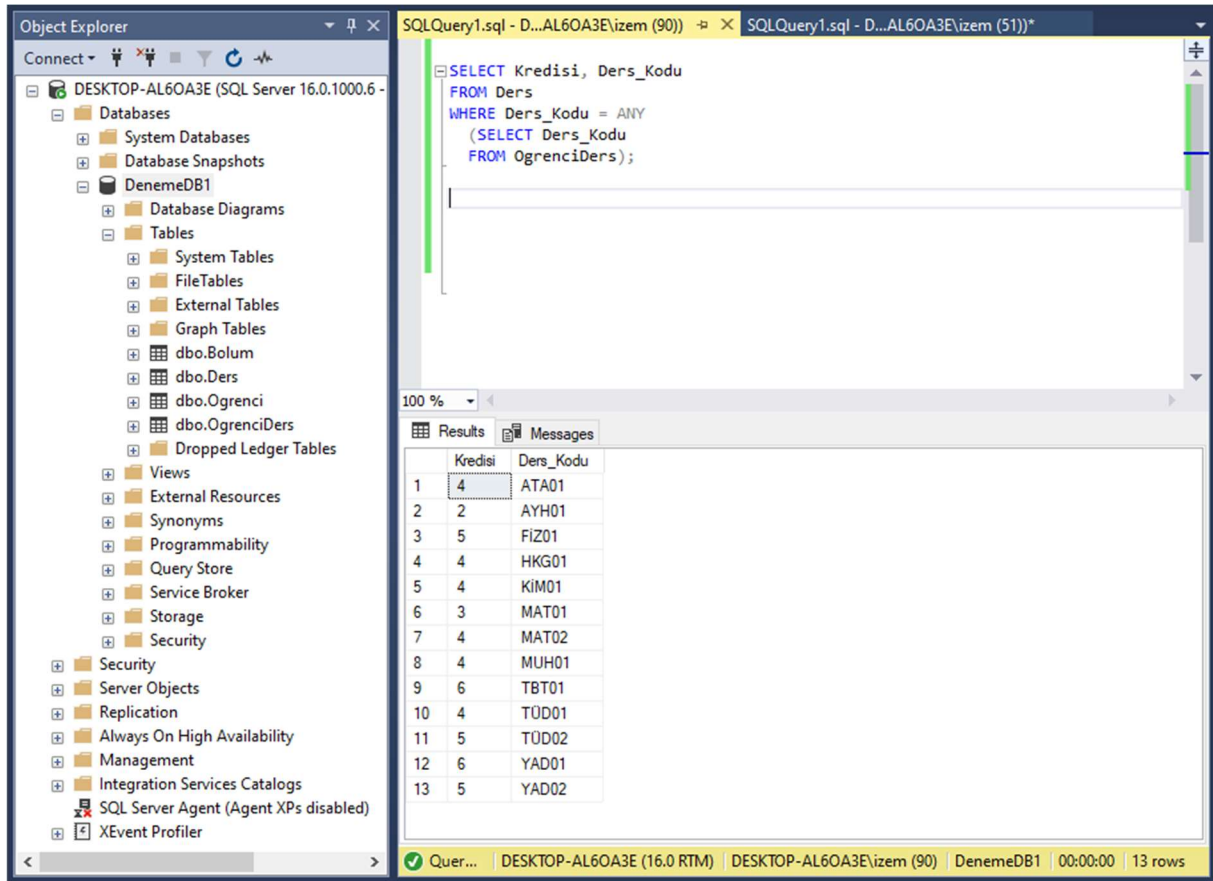
“ANY” ve “ALL” operatörleriyle de birlikte kullanılabilen iç içe “SELECT” ifadesi, tek bir kolon değeri ile bir dizi başka değer arasında karşılaştırma yapılmasını sağlar.

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ANY
(SELECT column_name
FROM table_name
WHERE condition);
```

Bir tablo adını, kolon isimlerini ve “operator” olarak “=, <>, !=, >, >=, <, ya da <=” ifadelerini parametre olarak alır. Alt sorgudaki koşullardan herhangi biri sağlanırsa, TRUE değeri döndürülür. Diğer bir ifadeyle “ANY”, aralıktaki değerlerden herhangi biri için işlem doğruysa koşulun doğru olacağı anlamına gelir.

Aşağıdaki “ANY” operatörüyle kombinlenen iç içe SELECT sorgusu, bir önceki “IN” operatörüyle kombinlenen iç içe SELECT sorgusuyla aynı sonucu döndürür. Çünkü “IN” ifadesi de “ANY” ifadesi de bir tür “OR” işlevi görmektedir.

```
SELECT Kredisi, Ders_Kodu
FROM Ders
WHERE Ders_Kodu = ANY
(SELECT Ders_Kodu
FROM OgrenciDers);
```

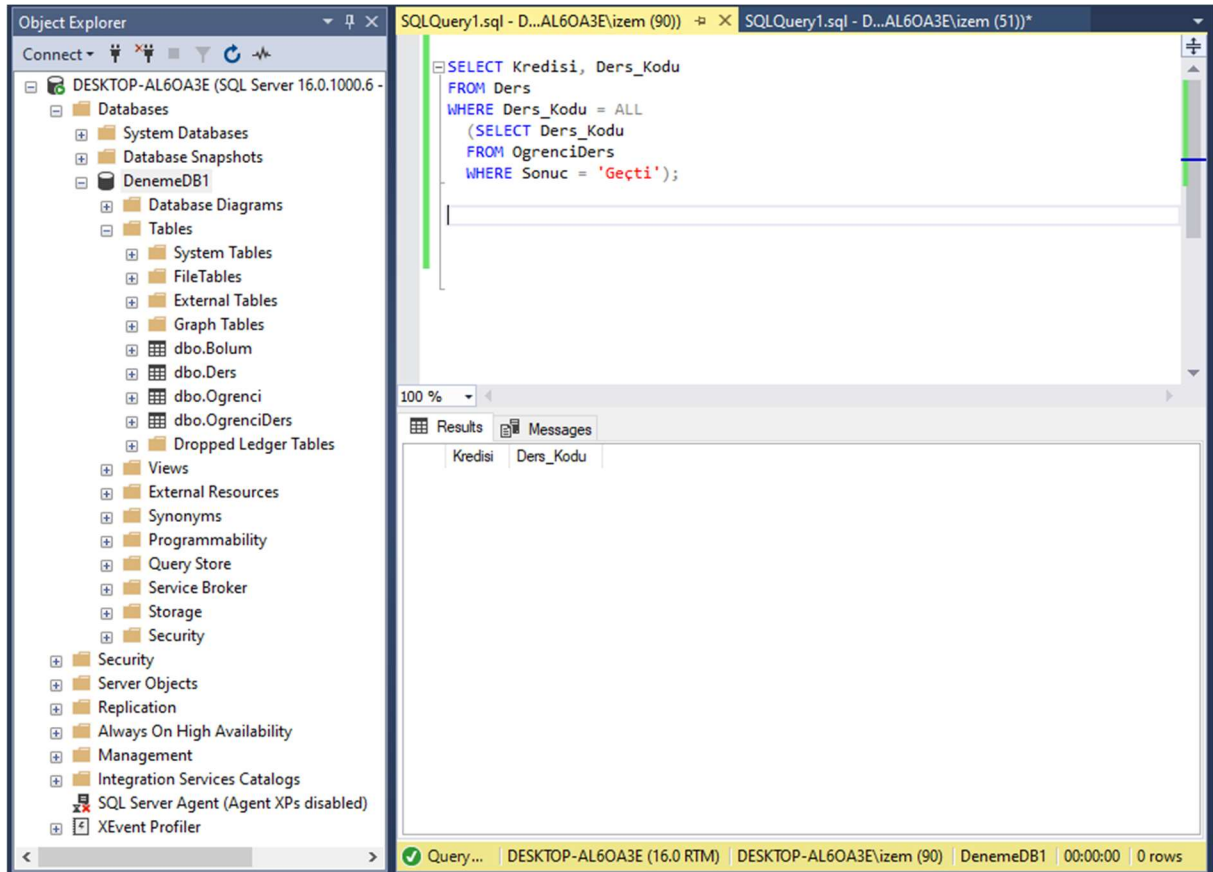


```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ALL
(SELECT column_name
FROM table_name
WHERE condition);
```

Bir tablo adını, kolon isimlerini ve “operator” olarak “=, <, !=, >, >=, <, ya da <=” ifadelerini parametre olarak alır. Alt sorgudaki koşulların tümü sağlanırsa, TRUE değeri döndürülür. Diğer bir ifadeyle “ALL”, aralıktaki değerlerin tümü için işlem doğruysa koşulun doğru olacağı anlamına gelir.

Aşağıdaki sorguda, OgrenciDers tablosunda yer alan Sonuc kayıtlarının tümü “Geçti” değerine sahip olmadığından, “ALL” ile kombinlenen alt sorgu false değeri döndürür. Bu yüzden Ders tablosunun Ders_Kodu kolonunda herhangi bir kayıtle eşleşme sağlanamadığı için aşağıdaki sorgu herhangi bir değer döndürmemiştir.

```
SELECT Kredisi, Ders_Kodu
FROM Ders
WHERE Ders_Kodu = ALL
(SELECT Ders_Kodu
FROM OgrenciDers
WHERE Sonuc = 'Geçti');
```



14. Foreign Key (Yabancı Anahtar) Tanımlama (3 puan)

ALTER TABLE target_table

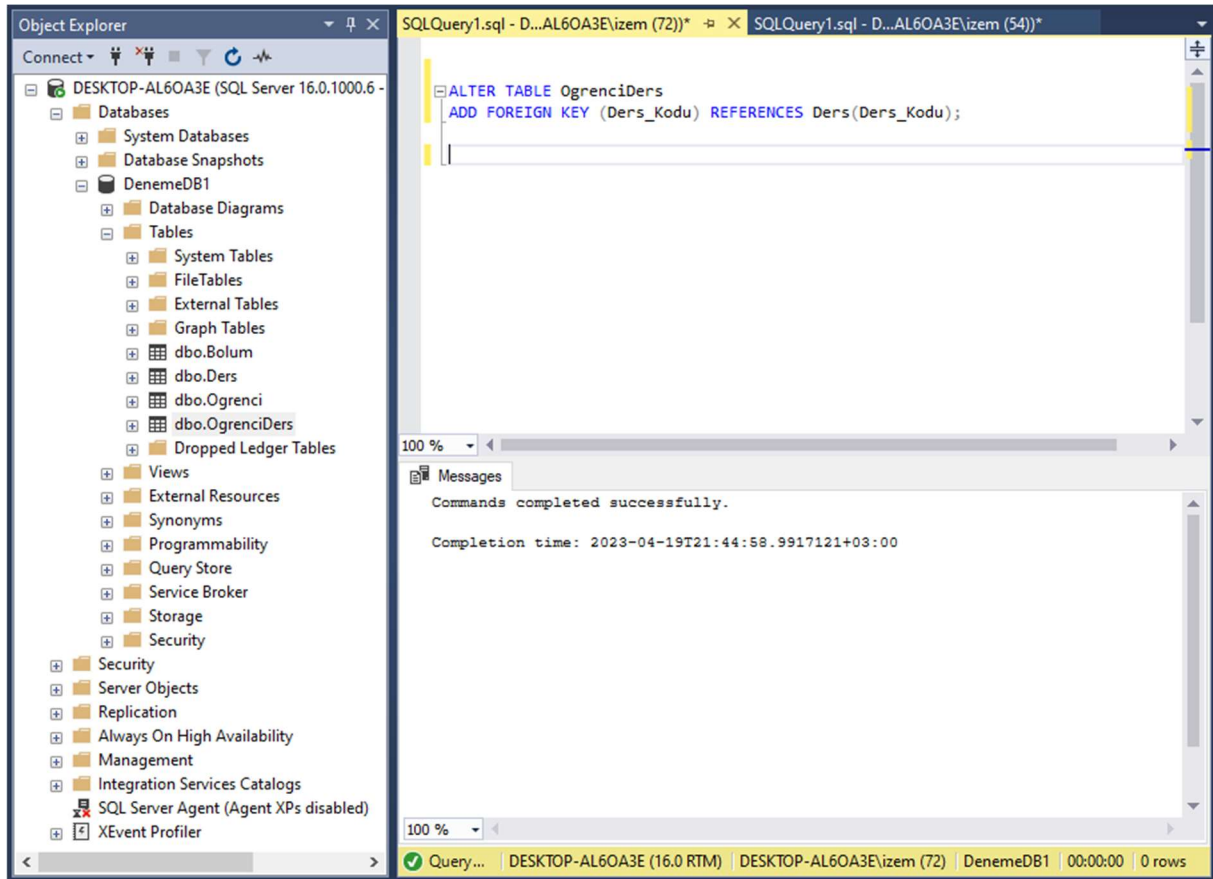
ADD FOREIGN KEY (target_column) REFERENCES source_table(source_column);

“FOREIGN KEY”, yani yabancı anahtar kısıtlaması, tablolar arasındaki bağlantılara zarar verecek eylemleri önlemek için kullanılır. Bir tablodaki yabancı anahtar, başka bir tablonun birincil anahtarına (primary key) işaret eder. Yabancı anahtarın yer aldığı tabloya çocuk tablo (child table), birincil anahtarın bulunduğu tabloya ise ebeveyn tablo (parent table) adı verilir. “FOREIGN KEY” ifadesi, kaynak tablo ile kaynak kolonu veyahut hedef tablo ile hedef kolonu parametre olarak alır.

Aşağıdaki sorguda, Ders tablosunun primary key’i olan Ders_Kodu kolonunun kayıtları referans alınarak, OgrenciDers tablosunun Ders_Kodu isimli kolonunun kayıtları yabancı anahtar (foreign key) olarak atanmıştır.

ALTER TABLE OgrenciDers

ADD FOREIGN KEY (Ders_Kodu) REFERENCES Ders(Ders_Kodu);



15. INSERT İfadesi (3 puan)

*INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);*

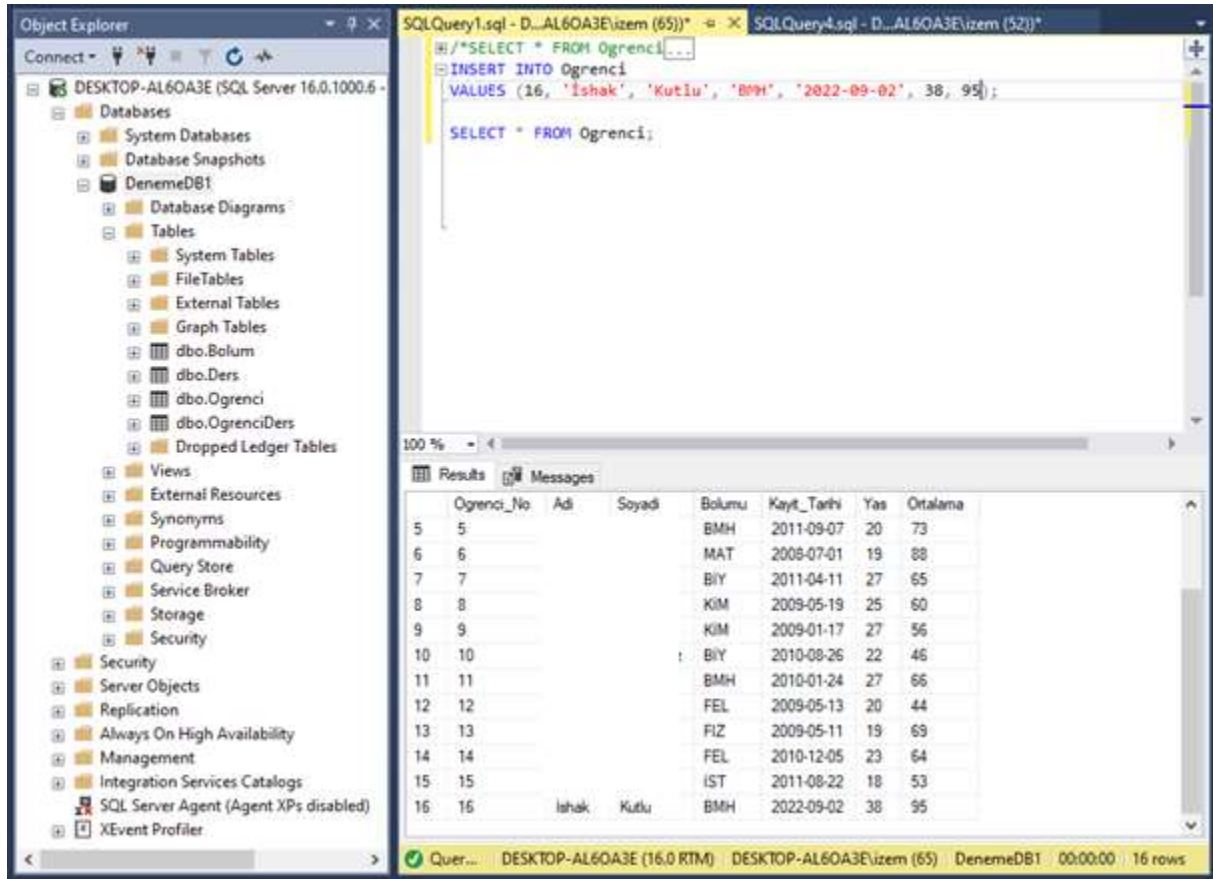
“INSERT INTO” ifadesi, tablo adı ile kolon isimlerini ve bunlara karşılık gelen değerleri parametre olarak alır. Belirli bir kolona kayıt eklemek için yukarıda gösterilen sentakstan yararlanılır.

*INSERT INTO table_name
VALUES (value1, value2, value3, ...);*

Belirli bir kolon yerine, tablonun tüm kolonlarını kayıt eklemek için yukarıda gösterilen sektakstan yararlanılır. Ancak “VALUES” kısmında yer alan parametre sayısı ekleme yapılmak istenen tabloda yer alan kolon sayısı ile aynı olmalıdır.

Aşağıdaki sorguyla Ogrenci tablosuna İshak Kutlu isimli yeni bir kayıt eklenmiştir.

*INSERT INTO Ogrenci
VALUES (16, 'İshak', 'Kutlu', 'BMH', '2022-09-02', 38, 95);
SELECT * FROM Ogrenci;*



16. INSERT INTO-SELECT İfadesi (3 Puan)

INSERT INTO table2

*SELECT * FROM table1;*

Bir kaynak tablodan bir hedef tabloya tüm kolonların kopyalandığı yukarıdaki sentaks, kaynak ve hedef tablo isimlerini parametre olarak alır.

INSERT INTO table2 (column1, column2, column3, ...)

SELECT column1, column2, column3, ...

FROM table1;

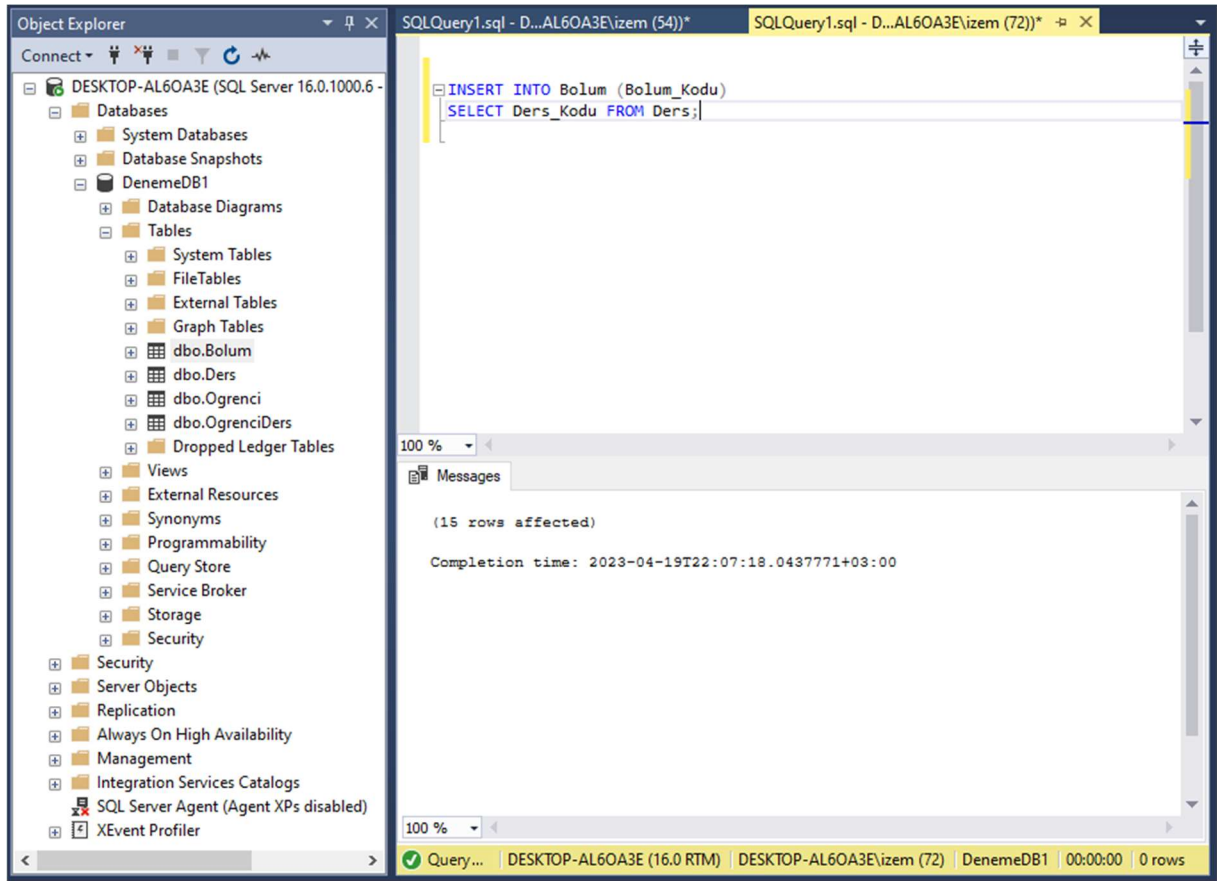
Yukarıdaki sentaksta ise kaynak tabloda yer alan belirli kolonlar, hedef tabloya kopyalanır. Bir önceki sentaksın parametrelerine ilave olarak her iki tablonun kolon isimlerini parametre olarak alır.

“INSERT INTO SELECT” ifadesi, bir tablodaki verileri kopyalar ve başka bir tabloya yerleştirir. Hedef tablodaki mevcut veriler bu işlemten etkilenmez. Ancak kaynak tablonun veri tipi ile hedef tablonun veri tipinin eşleşmesi gerekir.

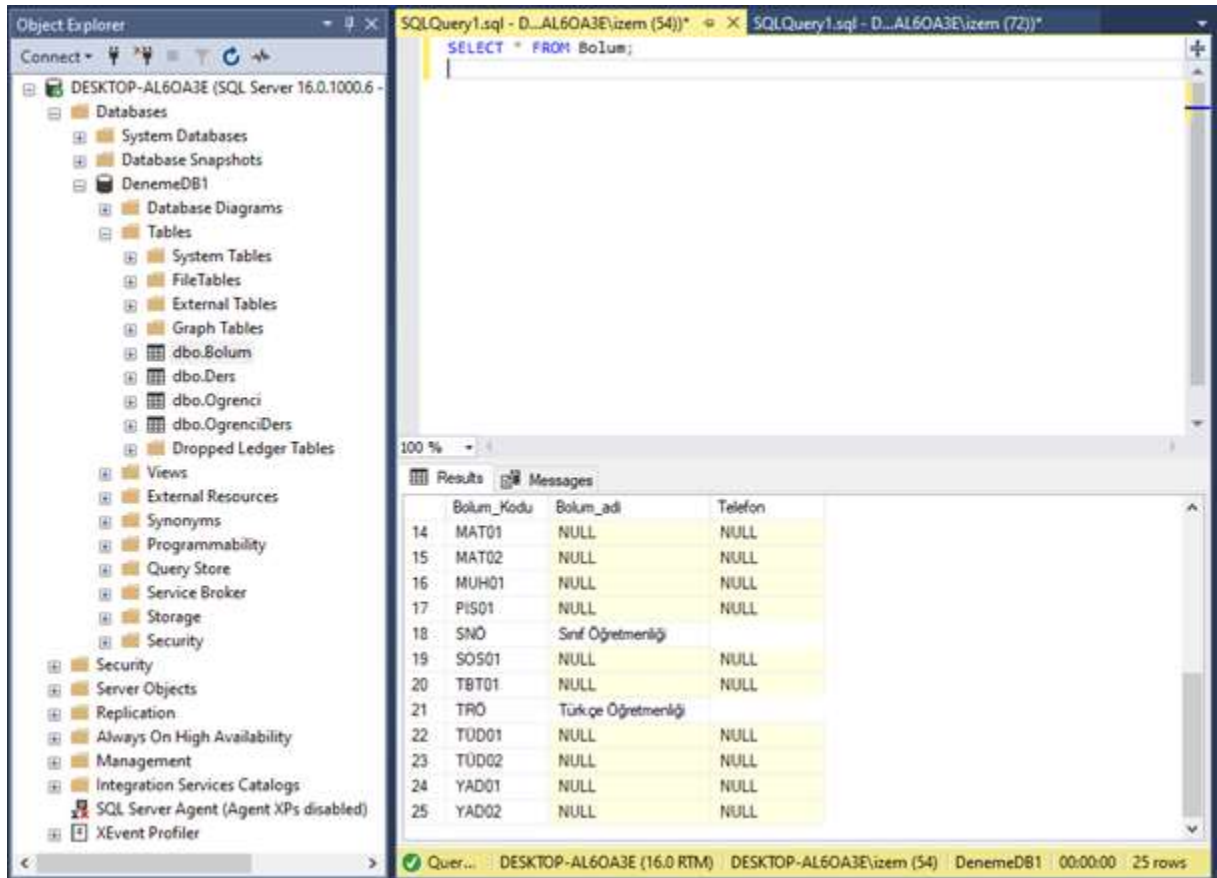
Aşağıdaki sorguda Ders tablosunun Ders_Kodu kolonu, Bolum tablosunun Bolum_Kodu kolonuna eklenmiştir. Bolum tablosunun diğer kolonları NULL değerlere sahip olacağı için tablo özelliklerinden ilgili kolonlar için NULL değerlere izin verilmiştir. Ayrıca kaynak ve hedef kolonun veri tipleri de birbiriyle aynı olacak şekilde ayarlandıktan sonra aşağıdaki sorgu başarılı bir şekilde çalıştırılmıştır.

INSERT INTO Bolum (Bolum_Kodu)

SELECT Ders_Kodu FROM Ders;



Aşağıdaki ekran görüntüsünde ise etkilenen Bolum tablosunun görünümüne yer verilmiştir.



17. UPDATE İfadesi (3 puan)

UPDATE table_name

SET column1 = value1, column2 = value2, ...

WHERE condition;

“UPDATE” ifadesi parametre olarak tablo adı ile kolon ismi ve değeri parametre olarak alır. Genellikle “WHERE” ifadesiyle kombinlenerek kullanılır. Bir tabloda yer alan mevcut bir kaydı değiştirmek amacıyla kullanılır.

Aşağıdaki sorguyla, Ogrenci tablosunda yer alan İshak Kutlu isimli öğrencinin ortalama notu 100 olarak güncellenmiş ve sonra da sonuç gösterilmiştir.

UPDATE Ogrenci

SET Ortalama = 100

WHERE Adi='İshak' AND Soyadi='Kutlu';

*SELECT * FROM Ogrenci;*

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL60A3E (SQL Server 16.0.1000.6)'. The 'Tables' folder is expanded, showing 'dbo.Ogrenci'. On the right, the SQL Query window displays the following query:

```
/*SELECT * FROM Ogrenci;...
UPDATE Ogrenci
SET Ortalama = 100
WHERE Adi='İshak' AND Soyadi='Kutlu';
SELECT * FROM Ogrenci;
```

Below the query window, the 'Results' tab shows the data from the 'Ogrenci' table. The table has 16 rows. The 16th row, corresponding to 'İshak Kutlu', shows an average score of 100.

Ogrenci_No	Adi	Soyadi	Bolumu	Kayit_Tarihi	Yas	Ortalama
5	5		BMH	2011-09-07	20	73
6	6		MAT	2008-07-01	19	88
7	7		BIY	2011-04-11	27	65
8	8		KIM	2009-05-19	25	60
9	9		KIM	2009-01-17	27	56
10	10		BIY	2010-08-26	22	46
11	11		BMH	2010-01-24	27	66
12	12		FEL	2009-05-13	20	44
13	13		FIZ	2009-05-11	19	69
14	14		FEL	2010-12-05	23	64
15	15		IST	2011-08-22	18	53
16	16	İshak	BMH	2022-09-02	38	100

18. DELETE İfadesi (3 puan)

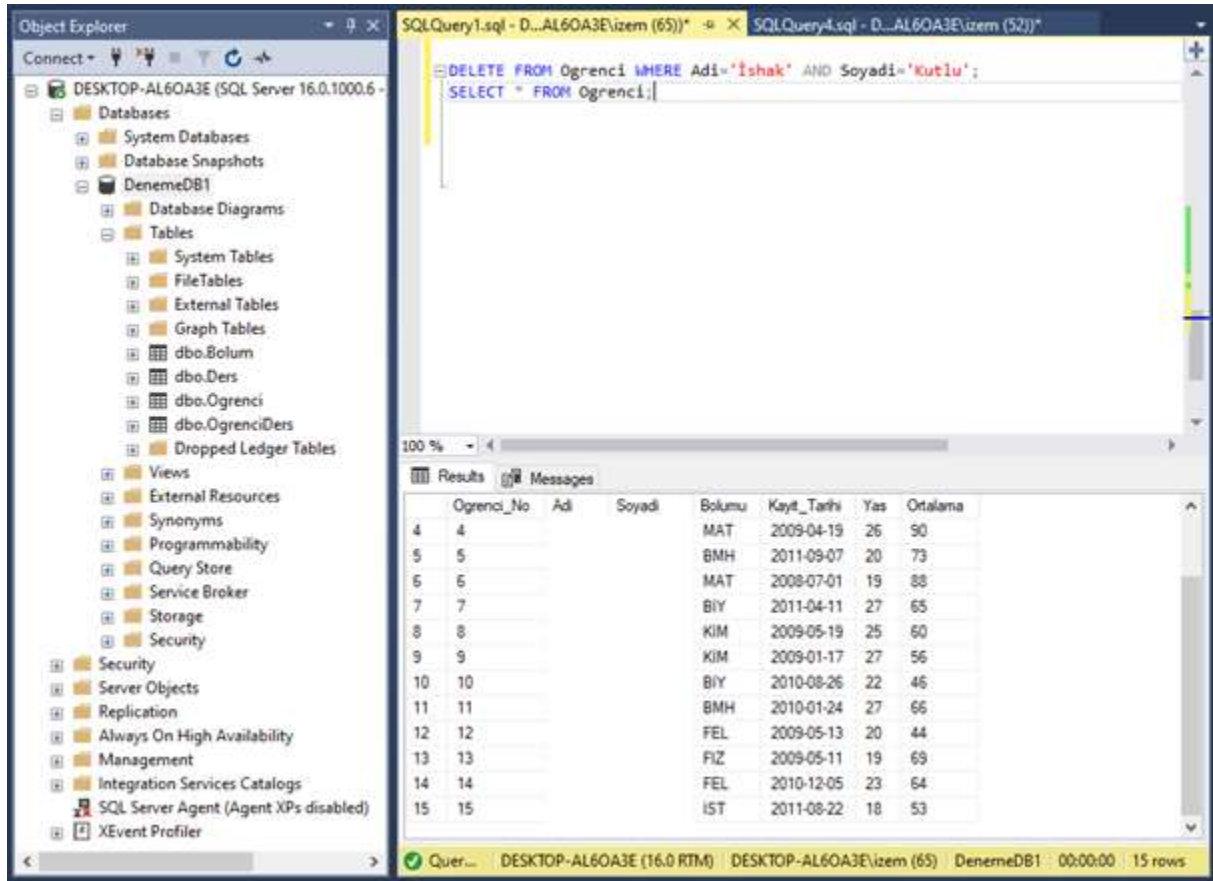
DELETE FROM table_name WHERE condition;

Parametre olarak sadece tablo ismini alan “DELETE” ifadesi, “WHERE” ifadesiyle kullanılarak bir tablodaki belirli bir kaydı siler. “WHERE” ifadesi olmadan kullanımı ise tablonun tamamının silinmesini ifade eder.

Aşağıdaki sorguyla Ogrenci tablosunda yer alan İshak Kutlu isimli kayıt silinmiş ve sonra da sonuç gösterilmiştir.

DELETE FROM Ogrenci WHERE Adi='İshak' AND Soyadi='Kutlu';

*SELECT * FROM Ogrenci;*



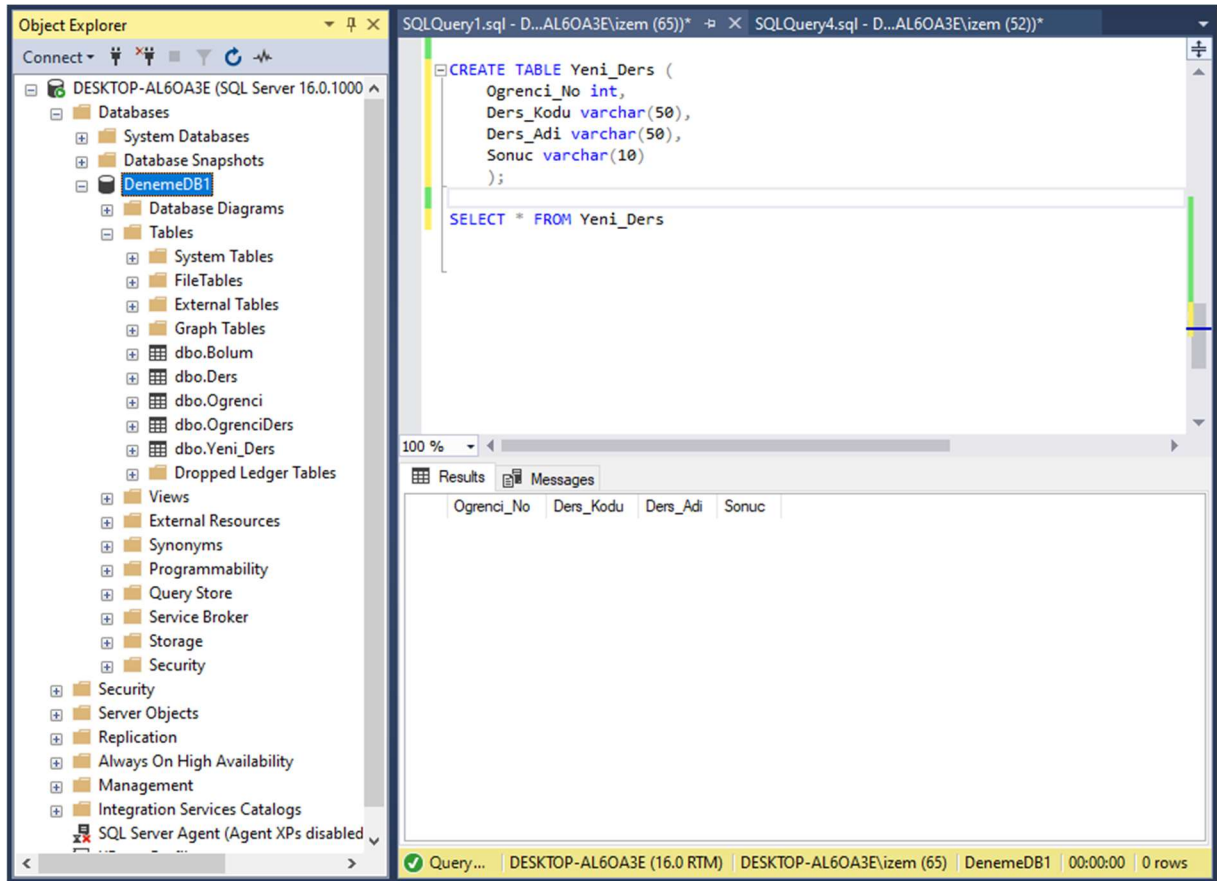
19. CREATE TABLE (3 puan)

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
    ....);
```

Parametre olarak tablo adı ile kolon isimlerini ve veri tiplerini alan “CREATE TABLE” ifadesi, veri tabanında yeni bir tablo oluşturulmasında kullanılır.

Aşağıdaki sorguda, int tipinde Ogrenci_No, varchar(50) tipinde Ders_Kodu, varchar(50) tipinde Ders_Adi ve varchar(10) tipinde Sonuc kolonlarından oluşan Yeni_Ders isimli bir tablo oluşturulmuş ve sonra da sonuç gösterilmiştir.

```
CREATE TABLE Yeni_Ders (
    Ogrenci_No int,
    Ders_Kodu varchar(50),
    Ders_Adi varchar(50),
    Sonuc varchar(10)
);
SELECT * FROM Yeni_Ders
```



20. ALTER TABLE (3 puan)

ALTER TABLE table_name

ADD/DROP/MODIFY/ALTER column_name datatype;

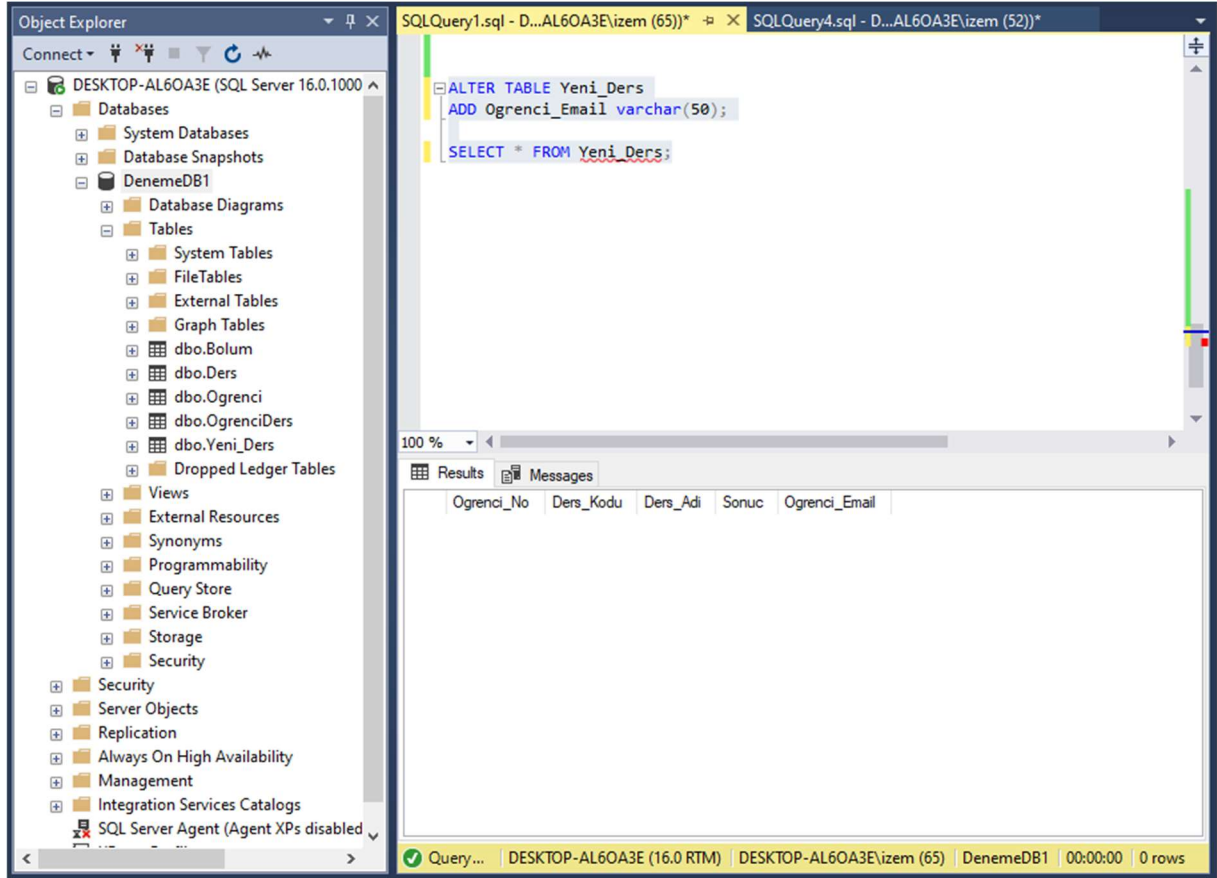
Parametre olarak tablo adı ile kolon adını ve veri tipini alan “ALTER TABLE” ifadesi, bir tablodaki kolonların değiştirilmesine, silinmesine ve yeni kolon eklenmesine olanak tanır.

Aşağıdaki sorguda Yeni_Ders isimli tabloya, varchar(50) tipinde Ogrenci_Email kolonu eklenmiş ve sonra da sonuç gösterilmiştir.

ALTER TABLE Yeni_Ders

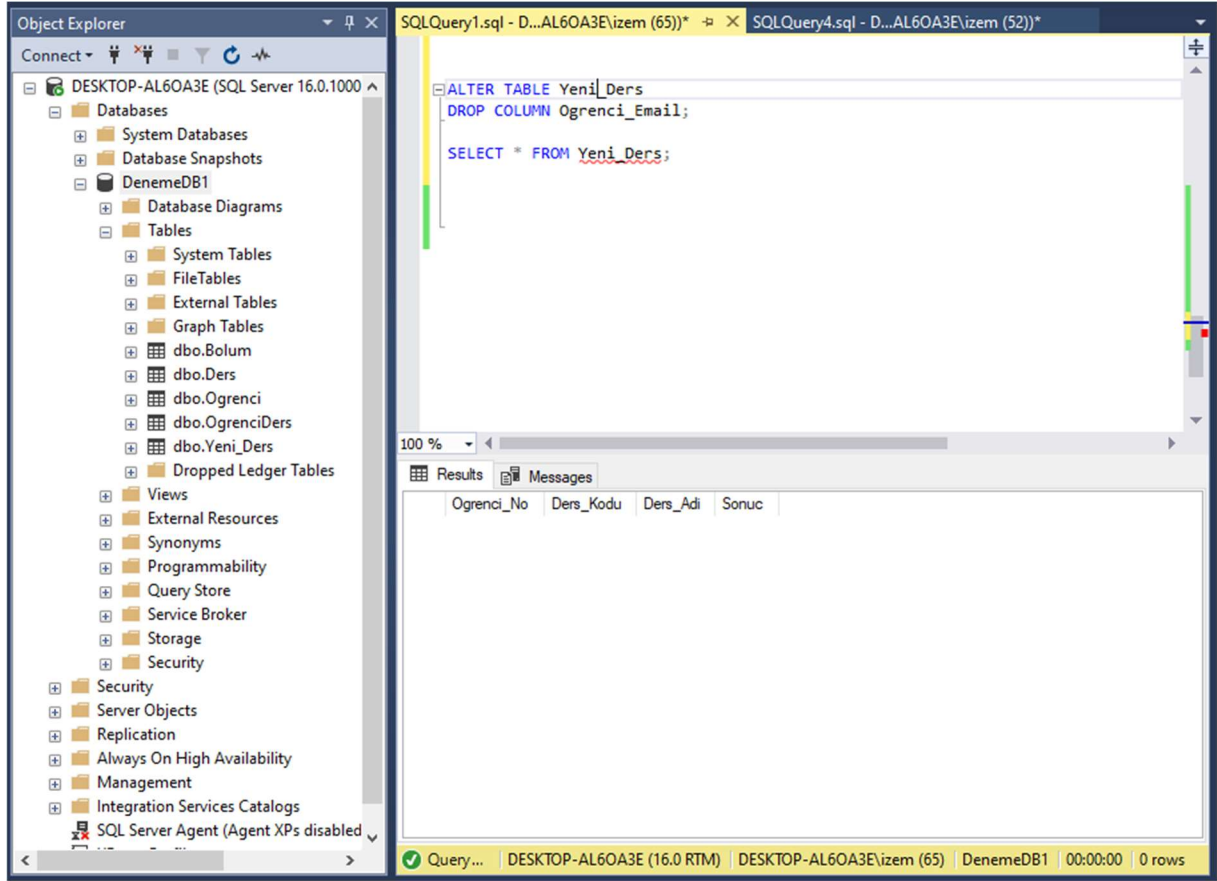
ADD Ogrenci_Email varchar(50);

*SELECT * FROM Yeni_Ders;*



Aşağıdaki sorguda, Yeni_Ders isimli tablodan Ogrenci_Email kolonu silinmiş ve sonra da sonuç gösterilmiştir.

```
ALTER TABLE Yeni_Ders
DROP COLUMN Ogrenci_Email;
SELECT * FROM Yeni_Ders;
```



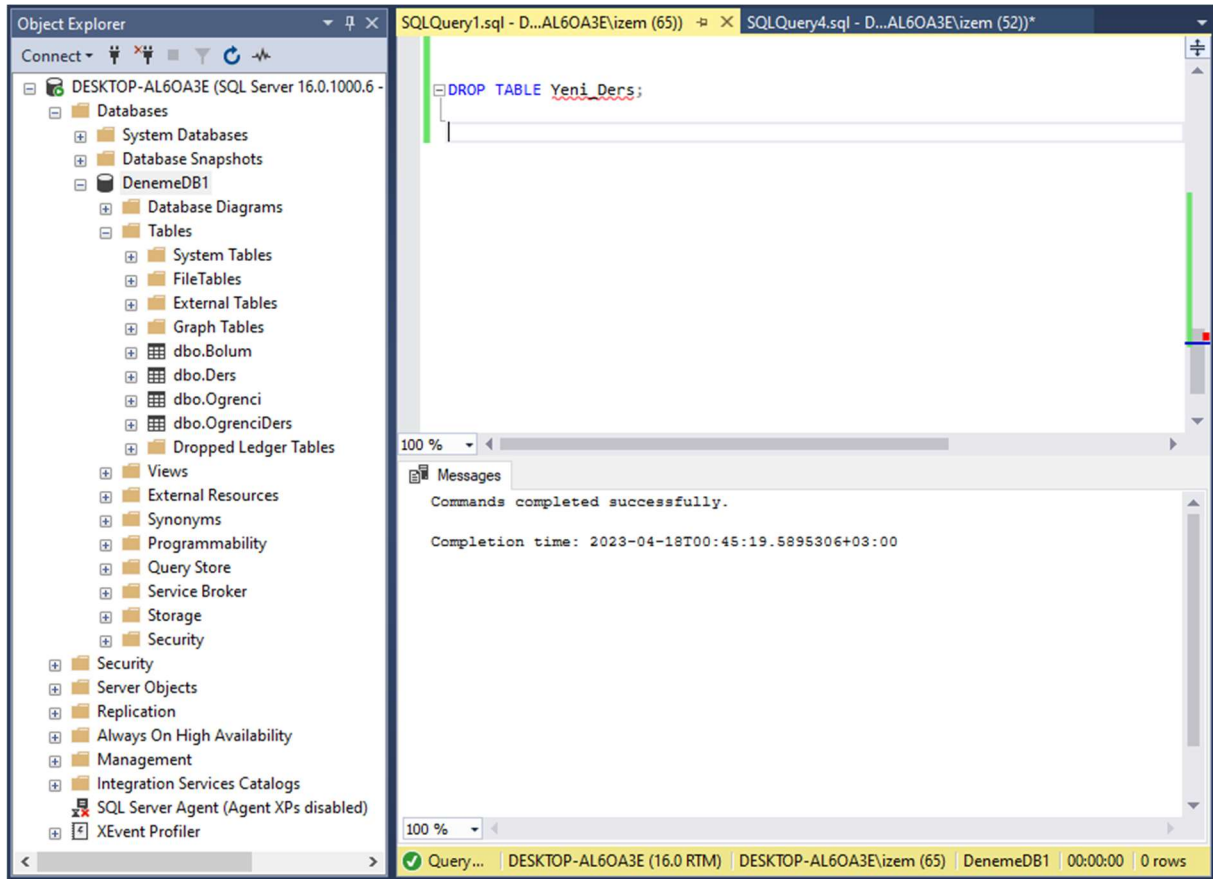
21. DROP TABLE İfadeleri (3 puan)

DROP TABLE table_name;

Sadece tablo adını parametre olarak alan “DROP TABLE” ifadesi, veri tabanında yer alan bir tablonun silinmesini sağlar.

Aşağıdaki sorguda, Yeni_Ders isimli tablo veri tabanından silinmiştir.

DROP TABLE Yeni_Ders;



22. CREATE PROCEDURE (İşlevi makul, mantıklı ve yeterli olacaktır. IF yapısı, CURSOR yapısı, Hata denetimi, transaction, eş zaman kontrolü içerecektir) (10 puan)

CREATE PROCEDURE işlevi için aşağıdaki değişkenler kullanılmıştır. Prosedür, yeni ya da var olan bir öğrencinin aldığı bir dersin ÖğrenciDers tablosuna eklenmesi ya da var olan bir kaydın değiştirilmesi gibi işlemleri yapmaya olanak tanıyacak şekilde tasarlanmıştır.

```
@Ogrenci_No int,  
@Ders_Kodu varchar(10),  
@Kac_Kez_Aldi int,  
@Vize_Notu int,  
@Final_Notu int,  
@Sonuc varchar(10)
```

IF yapıları kullanılarak ders kodunun 5 karakterden oluşması, eklenmek istene ders kodunun Ders_Kodu tablosunda yer alması, öğrenci no'nun Öğrenci tablosunda bulunması, vize ve final notunun 0 ile 100 arasında olması şeklinde kontroller eklenmiştir.

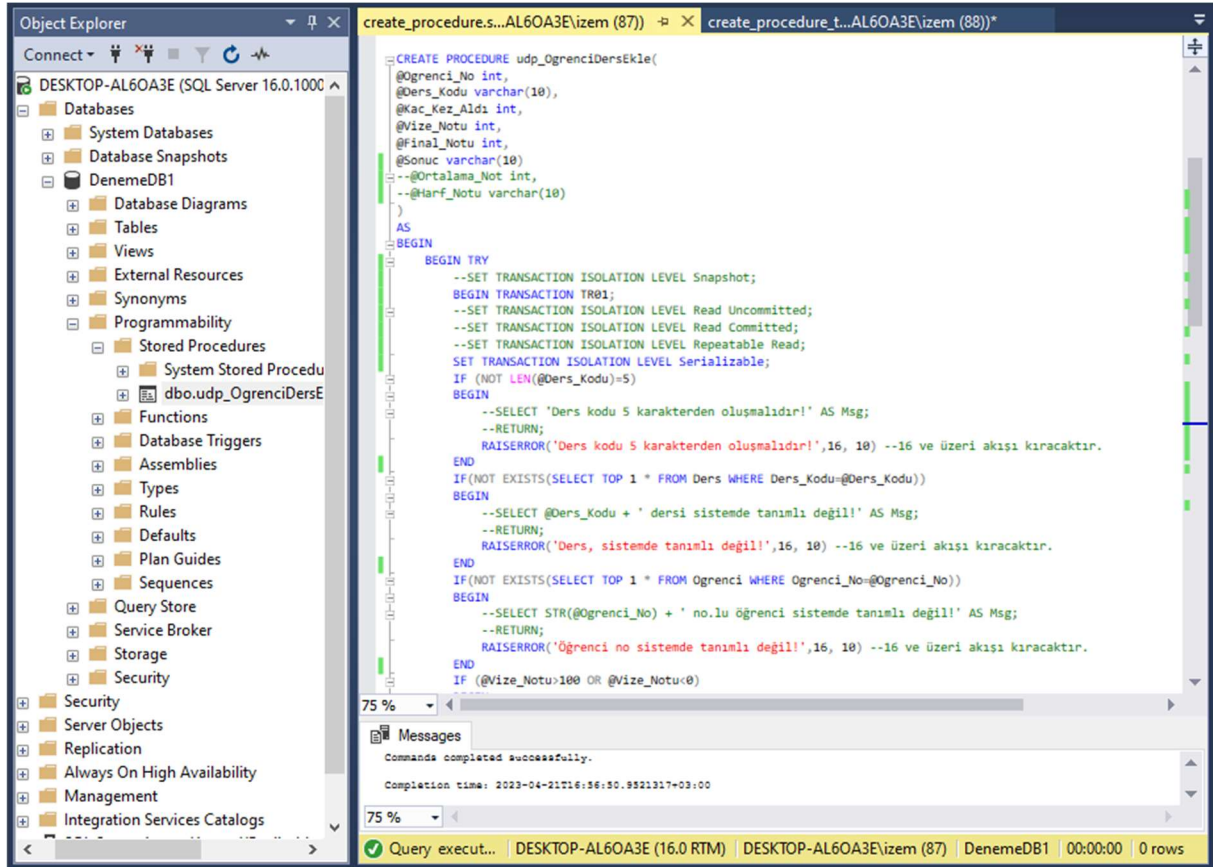
Benzer şekilde IF yapıları yardımıyla, bir dersi alan öğrencinin tekrar ÖğrenciDers tablosuna eklenmesi yerine var olan kaydın güncellenmesi, ÖğrenciDers tablosunda ders kodu ve öğrenci no'sunun tespit edilememesi durumunda ise yeni bir kaydın eklenmesi sağlanmıştır.

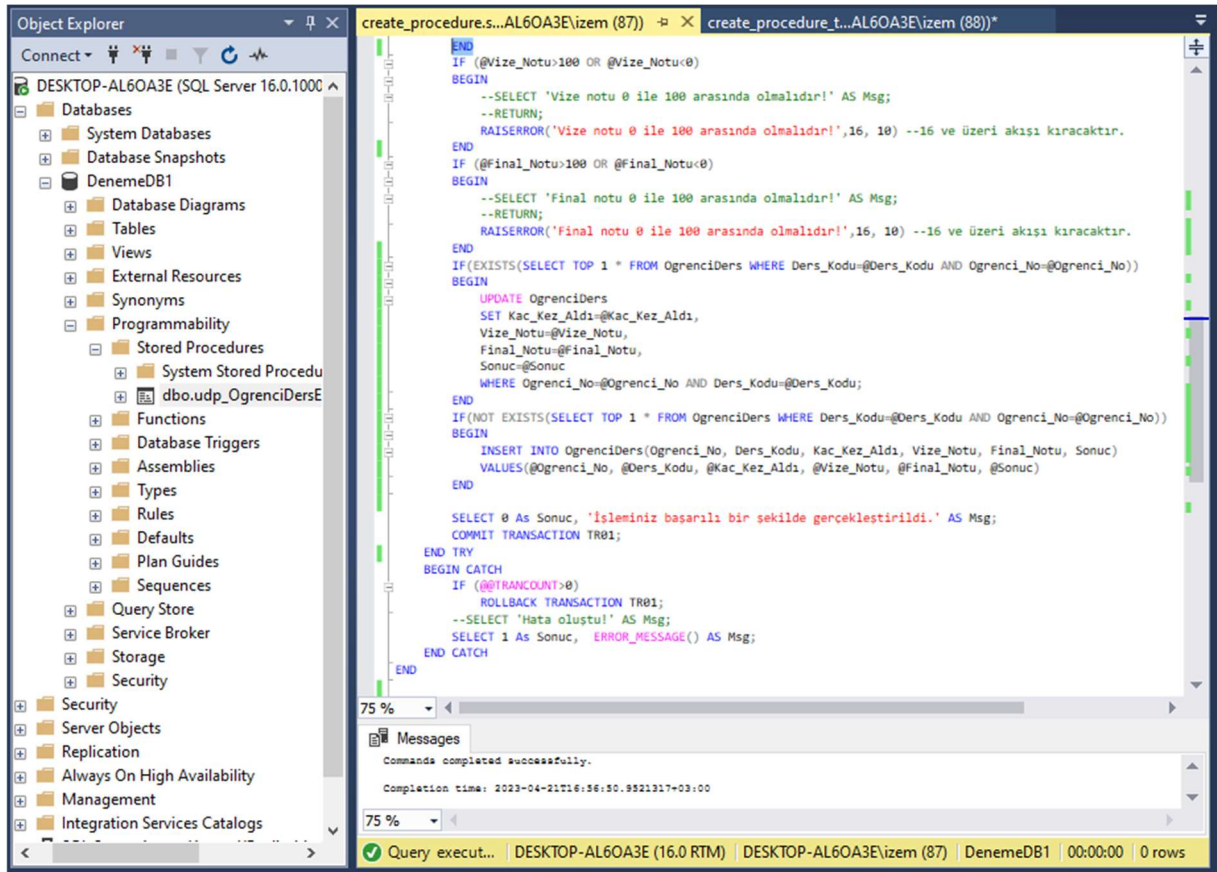
Hata denetimi için TRANSACTION-COMMIT-ROLLBACK ve TRY-CATCH bloklarından yararlanılmıştır. Gerçekleştirilecek işlemler veritabanına kalıcı olarak kaydedilmeden önce askıda bekletilmiş, herhangi bir hata durumunda try-catch blokları yardımıyla hatalar yakalanmış ve kullanıcıya hatayla ilgili mesaj gösterilmiştir. Problemsiz bir işlem commit ile veri tabanına kalıcı bir şekilde kaydedilmiş, hata durumunda ise kodun kırılımı sağlanarak

rollback yapılmış ve veri tabanı sorunlu işlemin öncesine döndürülerek veri tabanının tutarlılığı korunmuştur.

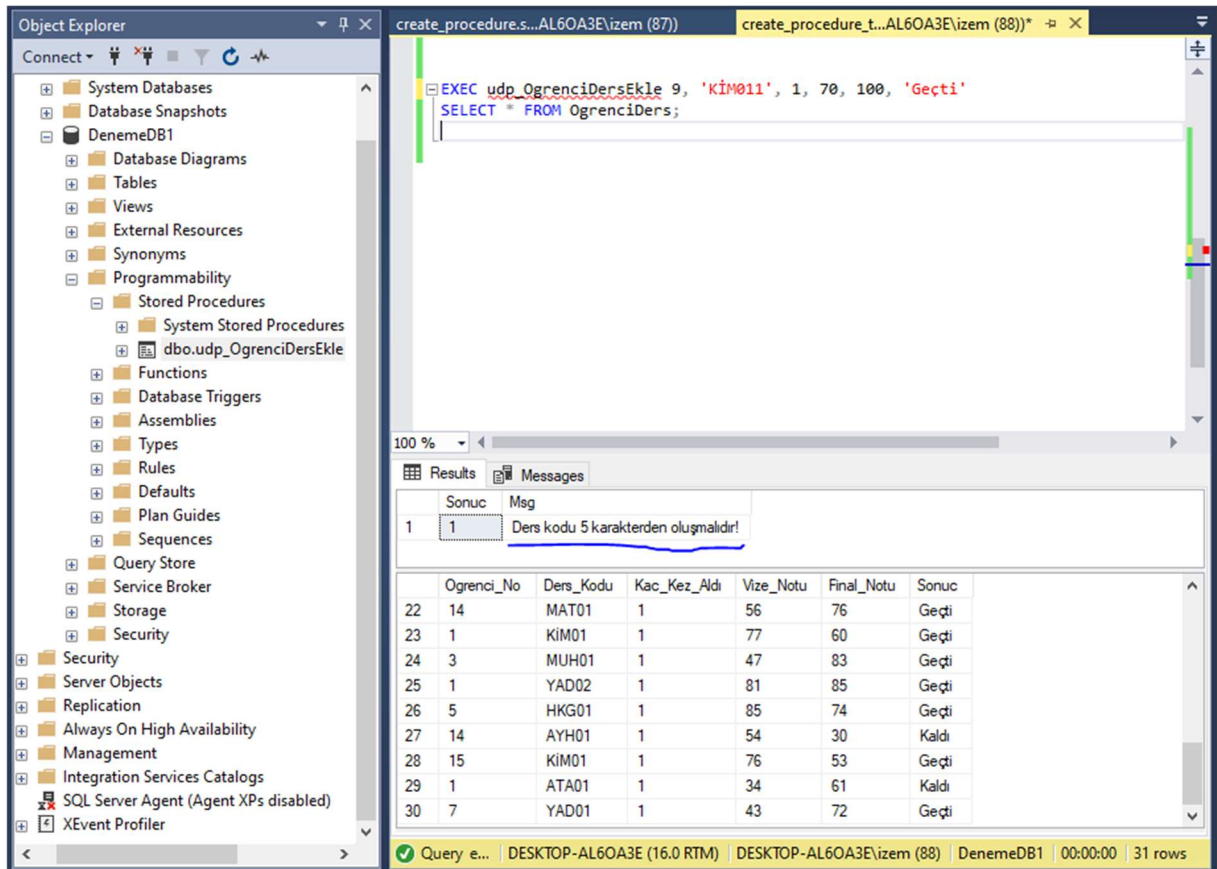
Eş zaman kontrolüyle ilgili çoklu kullanıcı desteğinde ortaya çıkabilecek bozuk okuma (dirty read), tekrarsız okuma (non-repeatable read) ve hayalet okuma (phantom read) problemlerine karşı, dördüncü seviye izolasyon seviyesi olan “Serializable” izolasyonundan yararlanılmıştır.

Aşağıdaki ekran görüntüsü CREATE PROCEDURE işlevi kullanılarak “udp_OgrenciDersEkle” isiminden kullanıcı tanımlı bir prosedür tanımlanmış ve çalıştırılmıştır.





Aşağıdaki ekran görüntüsünde, “KİM011” 6 karakterden oluşan bir ders kodu eklenmeye çalışılmış ve hata denetimi tarafından yakalanmıştır.



Aşağıdaki ekran görüntüsünde, Ders tablosunda yer almayan “İKT01” isimli bir ders kodu eklenmeye çalışılmış ve hata denetimi tarafından yakalanmıştır.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DenemeDB1'. The right pane shows a query window with the following SQL code:

```
EXEC udp_OgrenciDersEkle 9, 'İKT01', 1, 70, 100, 'Geçti'
SELECT * FROM OgrenciDers;
```

Below the query window, the Results pane displays an error message:

Sonuc	Msg
1	Ders, sistemde tanımlı değil!

Below the error message, a table of student records is displayed:

Ogrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	
22	14	MAT01	1	56	76	Geçti
23	1	KIM01	1	77	60	Geçti
24	3	MUH01	1	47	83	Geçti
25	1	YAD02	1	81	85	Geçti
26	5	HKG01	1	85	74	Geçti
27	14	AYH01	1	54	30	Kaldı
28	15	KIM01	1	76	53	Geçti
29	1	ATA01	1	34	61	Kaldı
30	7	YAD01	1	43	72	Geçti

The status bar at the bottom indicates the query was executed successfully, showing 31 rows.

Aşağıdaki ekran görüntüsünde, Ogrenci tablosunda yer almayan 16 no.lu bir öğrenciye ilişkin ders kayıtları eklenmeye çalışılmış ve işlem hata denetimi tarafından yakalanmıştır.

Object Explorer

Connect

System Databases

Database Snapshots

DenemeDB1

Database Diagrams

Tables

Views

External Resources

Synonyms

Programmability

Stored Procedures

System Stored Procedures

dbo.udp_OğrenciDersEkle

Functions

Database Triggers

Assemblies

Types

Rules

Defaults

Plan Guides

Sequences

Query Store

Service Broker

Storage

Security

Security

Server Objects

Replication

Always On High Availability

Management

Integration Services Catalogs

SQL Server Agent (Agent XPs disabled)

XEvent Profiler

create_procedure.s...AL60A3E\izem (87)

create_procedure_t...AL60A3E\izem (88)*

EXEC udp_OğrenciDersEkle 16, 'MUH01', 1, 70, 100, 'Geçti'

SELECT * FROM ÖğrenciDers;

Results

Messages

Sonuc

Msg

1 1 Öğrenci no sistemde tanımlı değil!

Oğrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	
1	9	FIZ01	1	64	55	Kaldı
2	8	MUH01	1	70	60	Geçti
3	11	AYH01	1	50	45	Kaldı
4	5	ATA01	1	63	38	Kaldı
5	9	TBT01	1	91	59	Geçti
6	3	MAT02	1	80	70	Geçti
7	12	ATA01	1	85	37	Kaldı
8	15	TUD01	1	25	84	Geçti
9	6	TUD02	1	30	30	Kaldı

Query e... DESKTOP-AL60A3E (16.0 RTM) DESKTOP-AL60A3E\izem (88) DenemeDB1 00:00:00 31 rows

Aşağıdaki ekran görüntüsünde, vize notu 150 olan kayıt eklenmeye çalışılmış ve işlem hata denetimi tarafından yakalanmıştır.

Object Explorer

Connect

System Databases

Database Snapshots

DenemeDB1

Database Diagrams

Tables

Views

External Resources

Synonyms

Programmability

Stored Procedures

System Stored Procedures

dbo.udp_OğrenciDersEkle

Functions

Database Triggers

Assemblies

Types

Rules

Defaults

Plan Guides

Sequences

Query Store

Service Broker

Storage

Security

Security

Server Objects

Replication

Always On High Availability

Management

Integration Services Catalogs

SQL Server Agent (Agent XPs disabled)

XEvent Profiler

create_procedure.s...AL60A3E\izem (87)

create_procedure_t...AL60A3E\izem (88)*

EXEC udp_OğrenciDersEkle 9, 'MAT01', 1, 150, 100, 'Geçti'

SELECT * FROM ÖğrenciDers;

Results

Messages

Sonuc

Msg

1 1 Vize notu 0 ile 100 arasında olmalıdır!

Oğrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	
1	9	FIZ01	1	64	55	Kaldı
2	8	MUH01	1	70	60	Geçti
3	11	AYH01	1	50	45	Kaldı
4	5	ATA01	1	63	38	Kaldı
5	9	TBT01	1	91	59	Geçti
6	3	MAT02	1	80	70	Geçti
7	12	ATA01	1	85	37	Kaldı
8	15	TUD01	1	25	84	Geçti
9	6	TUD02	1	30	30	Kaldı

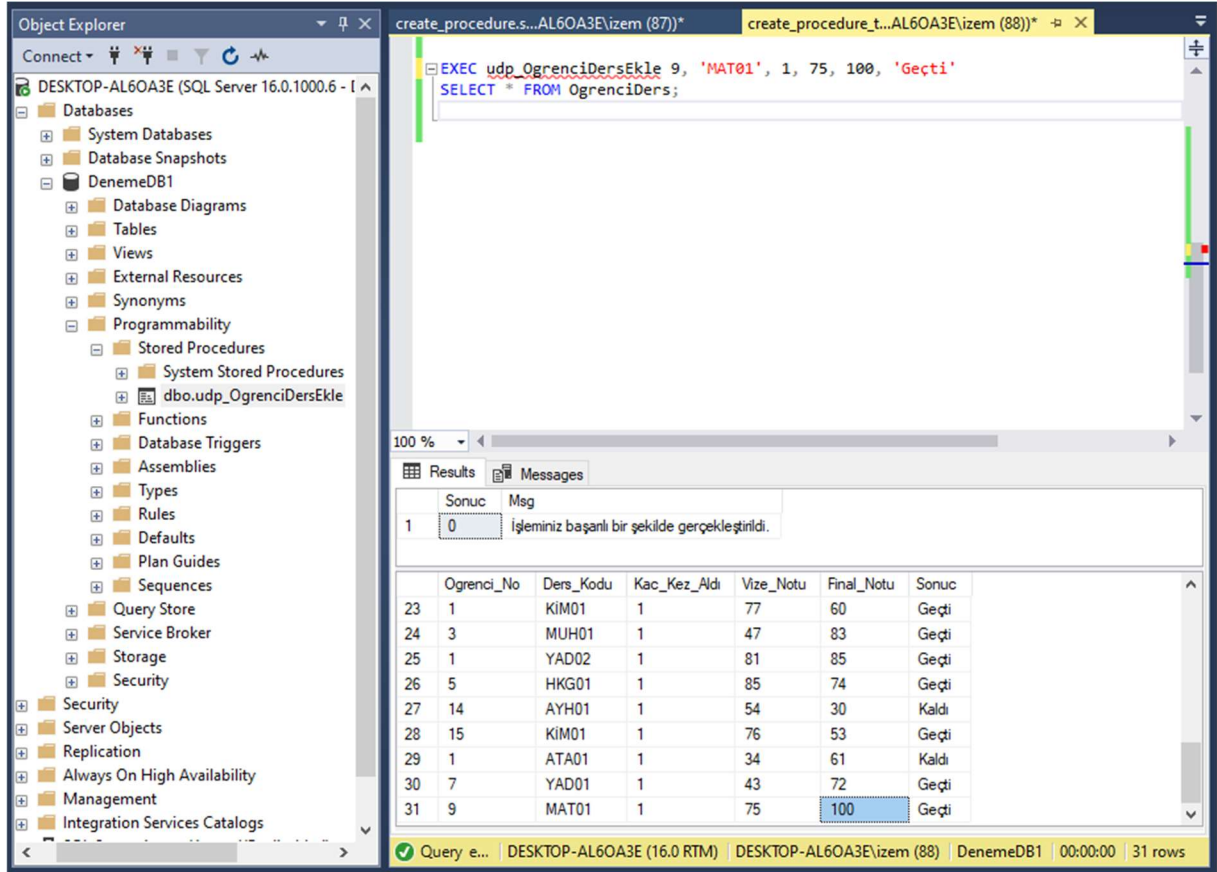
Query e... DESKTOP-AL60A3E (16.0 RTM) DESKTOP-AL60A3E\izem (88) DenemeDB1 00:00:00 31 rows

Aşağıdaki ekran görüntüsünde 9 no.lu öğrencinin “MAT01” dersine ilişkin dersi kaç kez aldığı, vize ve final notları ile sonucu başarılı bir şekilde ÖğrenciDers tablosuna eklenmiştir.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the database structure of 'DenemeDB1'. The right pane shows the execution of a stored procedure 'udp_OğrenciDersEkle' with parameters (9, 'MAT01', 1, 75, 90, 'Geçti'). The results pane displays a message indicating successful execution. Below the message, a table shows the data for the 'ÖğrenciDers' table.

Oğrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	
23	1	KIM01	1	77	60	Geçti
24	3	MUH01	1	47	83	Geçti
25	1	YAD02	1	81	85	Geçti
26	5	HKG01	1	85	74	Geçti
27	14	AYH01	1	54	30	Kaldı
28	15	KIM01	1	76	53	Geçti
29	1	ATA01	1	34	61	Kaldı
30	7	YAD01	1	43	72	Geçti
31	9	MAT01	1	75	90	Geçti

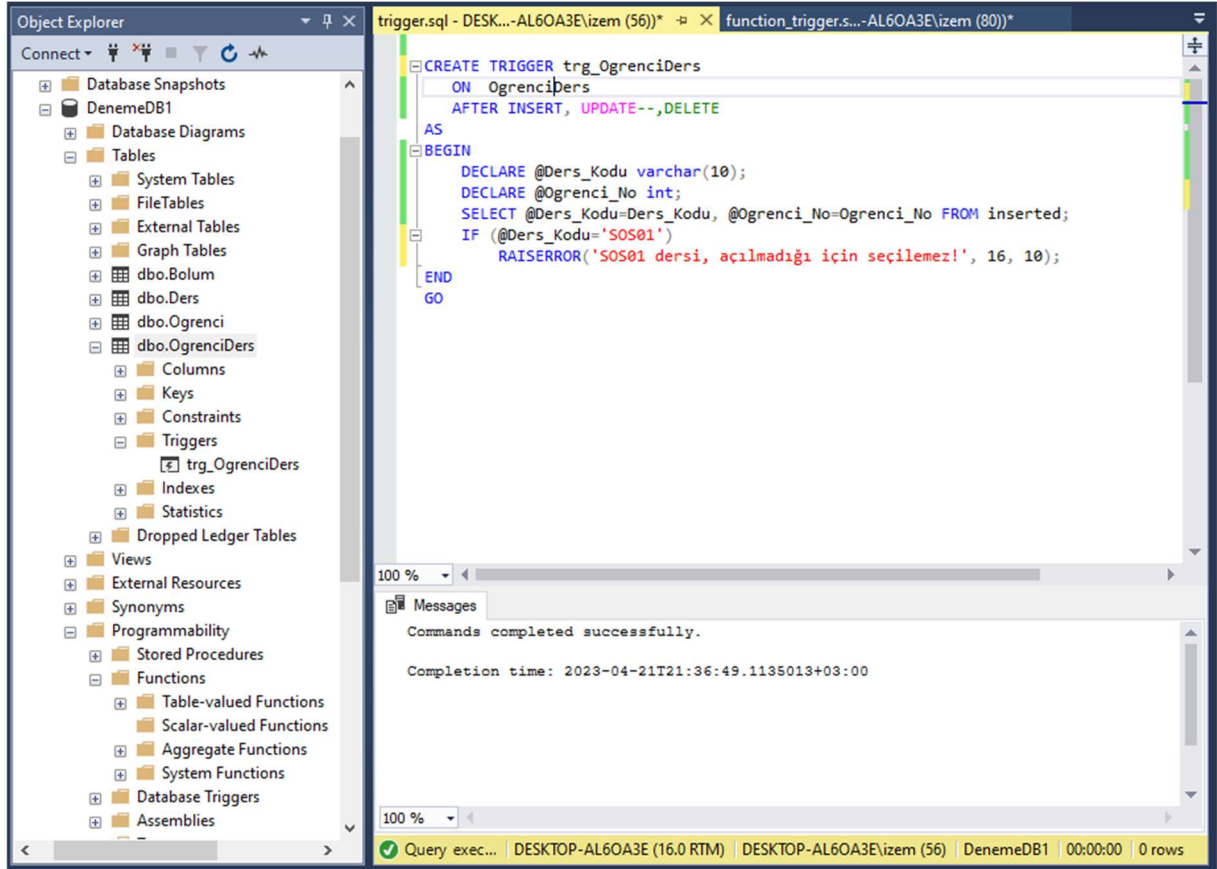
Aşağıdaki ekran görüntüsünde ise bir önceki aşamada 90 olarak kaydedilen final notu, 100 olarak güncellenmiştir.



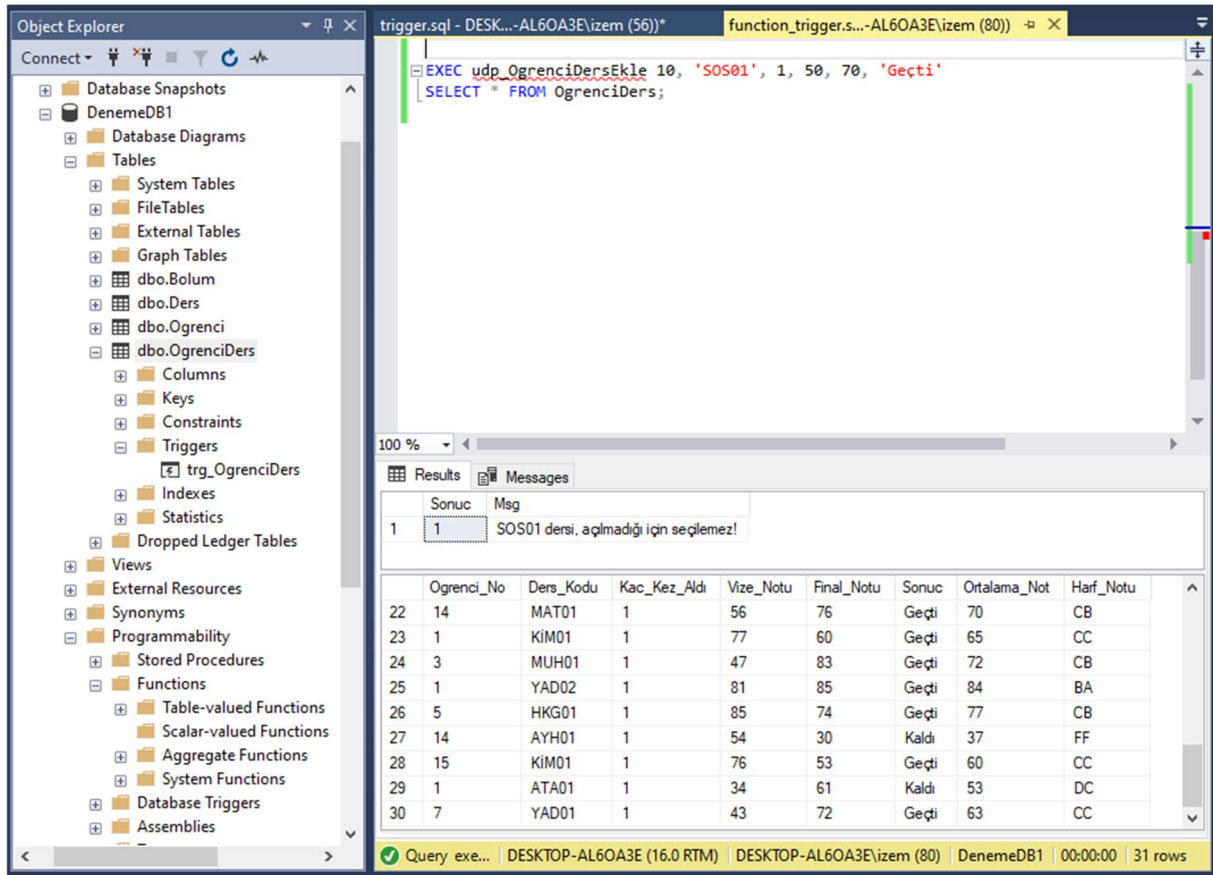
23. CREATE TRIGGER (İşlevi makul, mantıklı ve yeterli olacaktır, 7 puan)

CREATE TRIGGER işlevinin kullanım alanlarından biri olan RAISERROR hata kırılımını göstermek için Ders tablosunda yer almasına rağmen, 'SOS01' isimli dersin OgrenciDers tablosuna eklenmesine izin verilmeyecektir. İşlem kırıldığında ise 'SOS01 dersi, açılmadığı için seçilemez!' şeklinde bir bilgilendirme mesajı görüntülenecektir. Bu amaçla aşağıda gösterildiği gibi "trg_OgrenciDers" isimli bir trigger oluşturulmuştur.

OgrenciDers tablosuna her ne zaman insert ya da update yoluyla bir işlem yapılacak olursa, oluşturulan trigger otomatik olarak devreye girecektir.



Aşağıdaki ekran görüntüsünde, daha önce oluşturulan “udp_OgrenciDersEkle” isimli kullanıcı tanımlı prosedürün çalıştırılması sonrası, insert aşamasında eklenen kayıt, “trg_OgrenciDers” isimli triggeri başlatmış ve ‘SOS01’ dersinin yer aldığı kayıt kümesi tabloya işlenmeden ROLLBACK olmuştur.



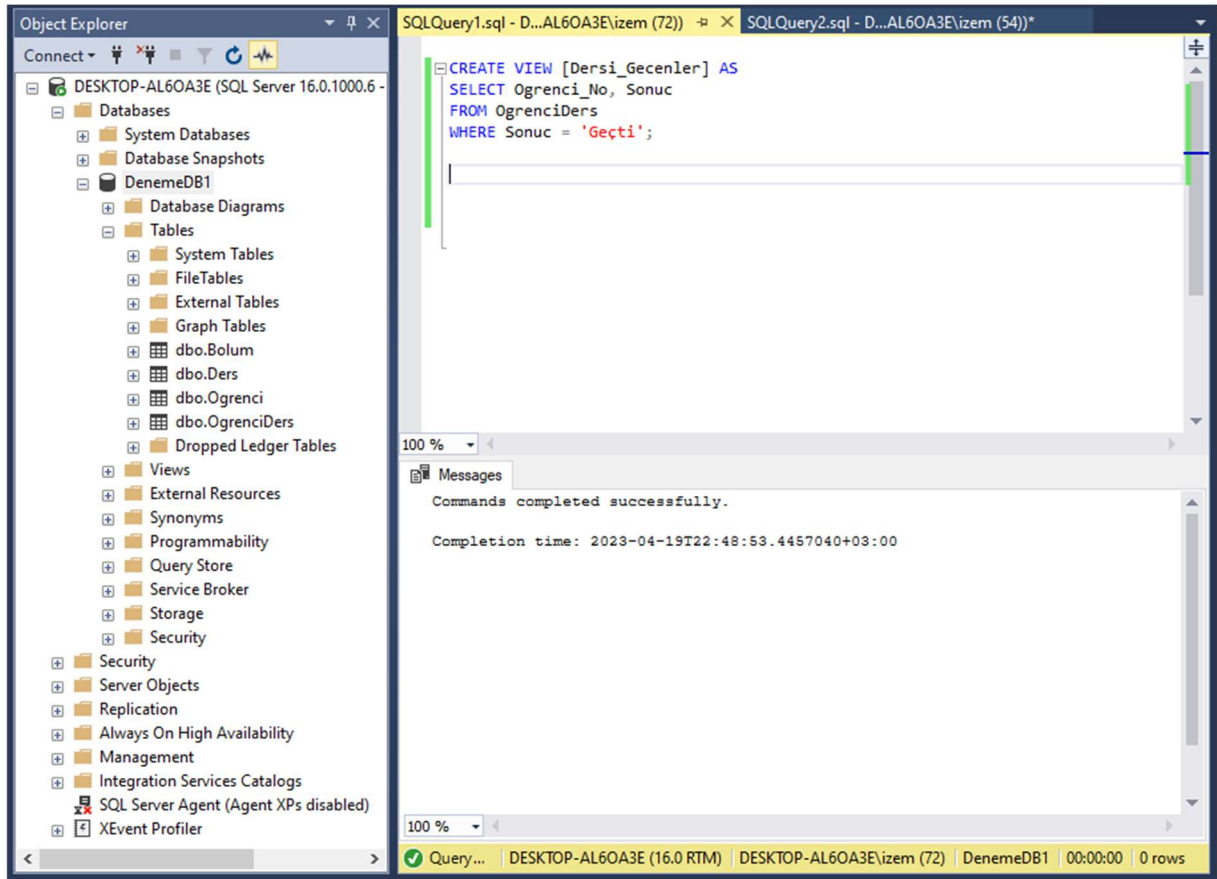
24. CREATE VIEW (3 puan)

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

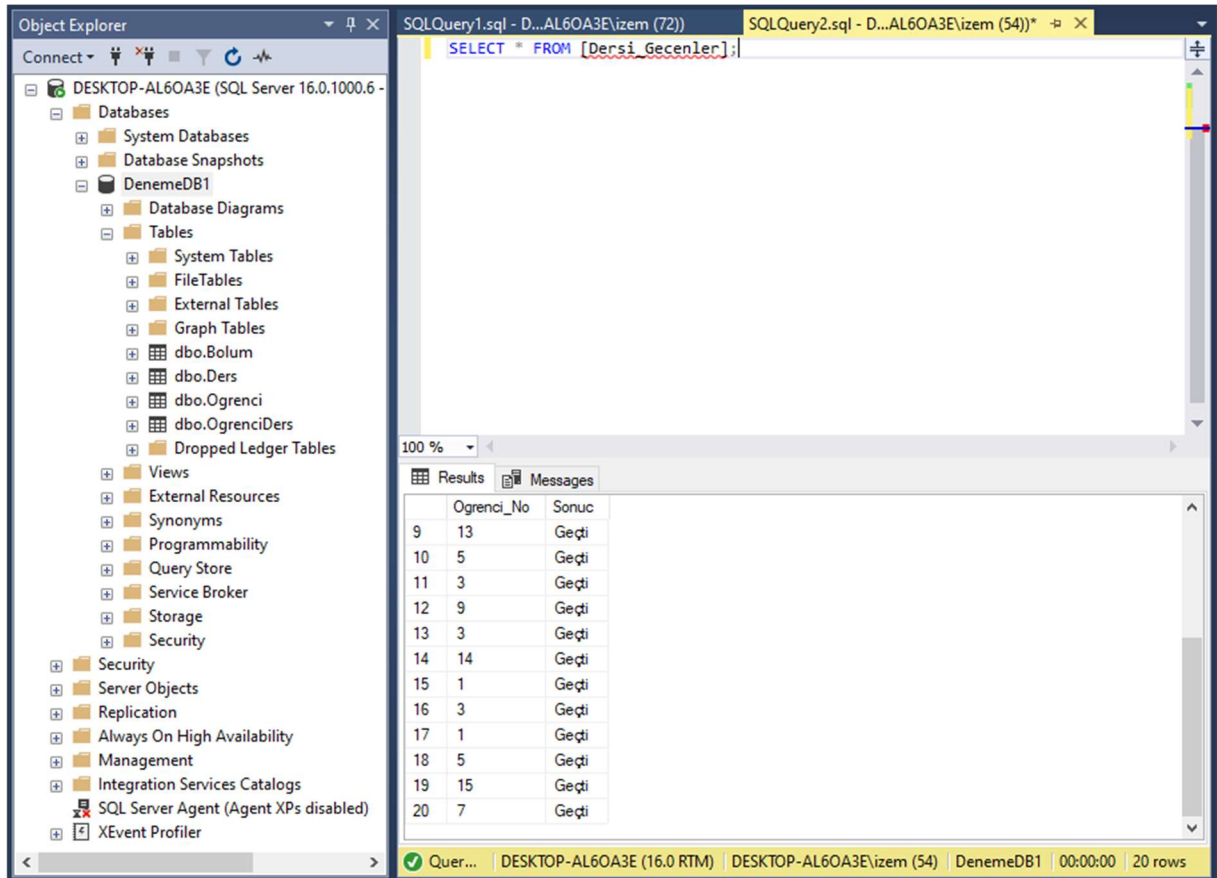
Parametre olarak tablo adını, kolon ve view (sanal tablo) ismini alan “CREATE VIEW” ifadesi, SQL sorgusuna dayalı sanal bir tablo oluşturulmasına olanak tanır. Sanal bir tablonun, gerçek bir tablodaki gibi kolonları ve satırları vardır. Sanal tablonun tüm kayıtları, bir veya daha fazla gerçek tablonun kayıtlarından oluşur.

Aşağıdaki sorguda, OgrenciDers tablosunun Ogrenci_No ve Sonuc kolonları kullanılarak Dersi_Gecenler isimli bir sanal tablo oluşturulmuştur.

```
CREATE VIEW [Dersi_Gecenler] AS
SELECT Ogrenci_No, Sonuc
FROM OgrenciDers
WHERE Sonuc = 'Geçti';
```



Aşağıdaki ekran görüntüsünde ise sanal tablo görüntülenmiştir.



25. CREATE FUNCTION (İşlevi makul, mantıklı ve yeterli olacaktır, 7 puan)

CREATE FUNCTION işlevini göstermek için ÖğrenciDers Tablosuna Ortalama_Not ve Harf_Notu isimli iki kolon eklenmiştir.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL6OA3E'. The central query window, titled 'create_procedure.s...AL6OA3E\izem (87)', contains the following SQL code:

```
ALTER TABLE ÖğrenciDers
ADD Ortalama_Not int, Harf_Notu varchar(10);
--DROP COLUMN Harf_Notu, Ortalama_Not;

--UPDATE ÖğrenciDers
--SET Ortalama_Not =CAST(ROUND((Vize_Notu)*0.3 + (Final_Notu)*0.7, 0) AS int)
SELECT * FROM ÖğrenciDers;
```

Below the query window, the 'Results' tab displays the data from the 'ÖğrenciDers' table. The table has 9 columns: Öğrenci_No, Ders_Kodu, Kac_Kez_Aldı, Vize_Notu, Final_Notu, Sonuc, Ortalama_Not, and Harf_Notu. The data is as follows:

Öğrenci_No	Ders_Kodu	Kac_Kez_Aldı	Vize_Notu	Final_Notu	Sonuc	Ortalama_Not	Harf_Notu
19	9	YAD02	1	98	38	Geçti	NULL
20	6	TBT01	1	43	61	Kaldı	NULL
21	3	YAD02	1	98	65	Geçti	NULL
22	14	MAT01	1	56	76	Geçti	NULL
23	1	KIM01	1	77	60	Geçti	NULL
24	3	MUH01	1	47	83	Geçti	NULL
25	1	YAD02	1	81	85	Geçti	NULL
26	5	HKG01	1	85	74	Geçti	NULL
27	14	AYH01	1	54	30	Kaldı	NULL
28	15	KIM01	1	76	53	Geçti	NULL
29	1	ATA01	1	34	61	Kaldı	NULL
30	7	YAD01	1	43	72	Geçti	NULL

The status bar at the bottom indicates 'Query executed successfully.' and shows the connection details: 'DESKTOP-AL6OA3E (16.0 RTM) | DESKTOP-AL6OA3E\izem (88) | DenemeDB1 | 00:00:00 | 30 rows'.

Ortalama_Not kolonunda Vize_Notu'nun yüzde 30'u ile Final_Notu'nun yüzde 70'i aşağıda gösterildiği gibi int değer döndürecek şekilde SQL komutuna hesaplatılmıştır.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the database structure of 'DenemeDB1'. The main query window shows a script for creating a procedure named 'create_procedure_t...AL60A3E\izem (88)'. The script includes SQL commands to alter a table, add a column, drop a column, and update a table. Below the script, the 'Results' tab displays a table with 30 rows of data.

Ogrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	Ortalama_Not	Harf_Notu
19	9	YAD02	1	38	Geçi	56	NULL
20	6	TBT01	1	61	Kaldı	56	NULL
21	3	YAD02	1	65	Geçi	75	NULL
22	14	MAT01	1	76	Geçi	70	NULL
23	1	KIM01	1	60	Geçi	65	NULL
24	3	MUH01	1	83	Geçi	72	NULL
25	1	YAD02	1	85	Geçi	84	NULL
26	5	HKG01	1	74	Geçi	77	NULL
27	14	AYH01	1	30	Kaldı	37	NULL
28	15	KIM01	1	53	Geçi	60	NULL
29	1	ATA01	1	61	Kaldı	53	NULL
30	7	YAD01	1	72	Geçi	63	NULL

The status bar at the bottom indicates: 'Query executed successfully. DESKTOP-AL60A3E (16.0 RTM) DESKTOP-AL60A3E\izem (88) DenemeDB1 00:00:00 30 rows'.

Harf_Notu kolonundaki kayıtları oluşturmak için CREATE FUNCTION işlevinden yararlanılacaktır. Bu amaçla, aşağıdaki ekran görüntüsünde gösterildiği gibi önce “udf_HarfNotu” isimli kullanıcı tanımlı bir fonksiyon oluşturulmuştur.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the database structure of 'DenemeDB1'. The main query window shows a script for creating a user-defined function named 'create_procedure_t...AL60A3E\izem (86)'. The script includes SQL commands to create a function, declare variables, and return a value. Below the script, the 'Messages' tab displays the completion time of the function creation.

```

--CREATE FUNCTION udf_HarfNotu(
--@Ortalama_Not int
--@Ogrenci_No int
--)
--RETURNS varchar(10)
--AS
--BEGIN
--DECLARE @Ortalama_Not int
--SELECT @Ortalama_Not=Ortalama_Not FROM OgrenciDers WHERE Ogrenci_No=@Ogrenci_No
--SET @Ortalama_Not = (SELECT TOP 1 Ortalama_Not FROM OgrenciDers WHERE Ogrenci_No=@Ogrenci_No)
--DECLARE @Harf_Notu varchar(10);
--SET @Harf_Notu=(
--CASE
--WHEN @Ortalama_Not>=90 AND @Ortalama_Not<=100 THEN 'AA'
--WHEN @Ortalama_Not>=80 AND @Ortalama_Not<90 THEN 'BA'
--WHEN @Ortalama_Not>=70 AND @Ortalama_Not<80 THEN 'CB'
--WHEN @Ortalama_Not>=60 AND @Ortalama_Not<70 THEN 'CC'
--WHEN @Ortalama_Not>=50 AND @Ortalama_Not<60 THEN 'DC'
--WHEN @Ortalama_Not>=40 AND @Ortalama_Not<50 THEN 'DD'
--ELSE 'FF'
--END);
--RETURN @Harf_Notu;
--*/
--END

```

The status bar at the bottom indicates: 'Query execut... DESKTOP-AL60A3E (16.0 RTM) DESKTOP-AL60A3E\izem (86) DenemeDB1 00:00:00 0 rows'.

Aşağıdaki gösterildiği gibi henüz Harf_Notu kolonundaki tüm kayıtlar “NULL” değerine sahiptir.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'DESKTOP-AL6OA3E (SQL Server 16.0.1000)'. The 'Programmability' folder is expanded, showing 'Functions' and 'Aggregate Functions'. The 'dbo.udf_HarfNotu' function is selected. The main window displays a query window with the following T-SQL code:

```
--ALTER TABLE OgrenciDers
--ADD Ortalama_Not int, Harf_Notu varchar(10);
--DROP COLUMN Harf_Notu, Ortalama_Not;

--UPDATE OgrenciDers
--SET Ortalama_Not =CAST(ROUND((Vize_Notu)*0.3 + (Final_Notu)*0.7, 0) AS int)

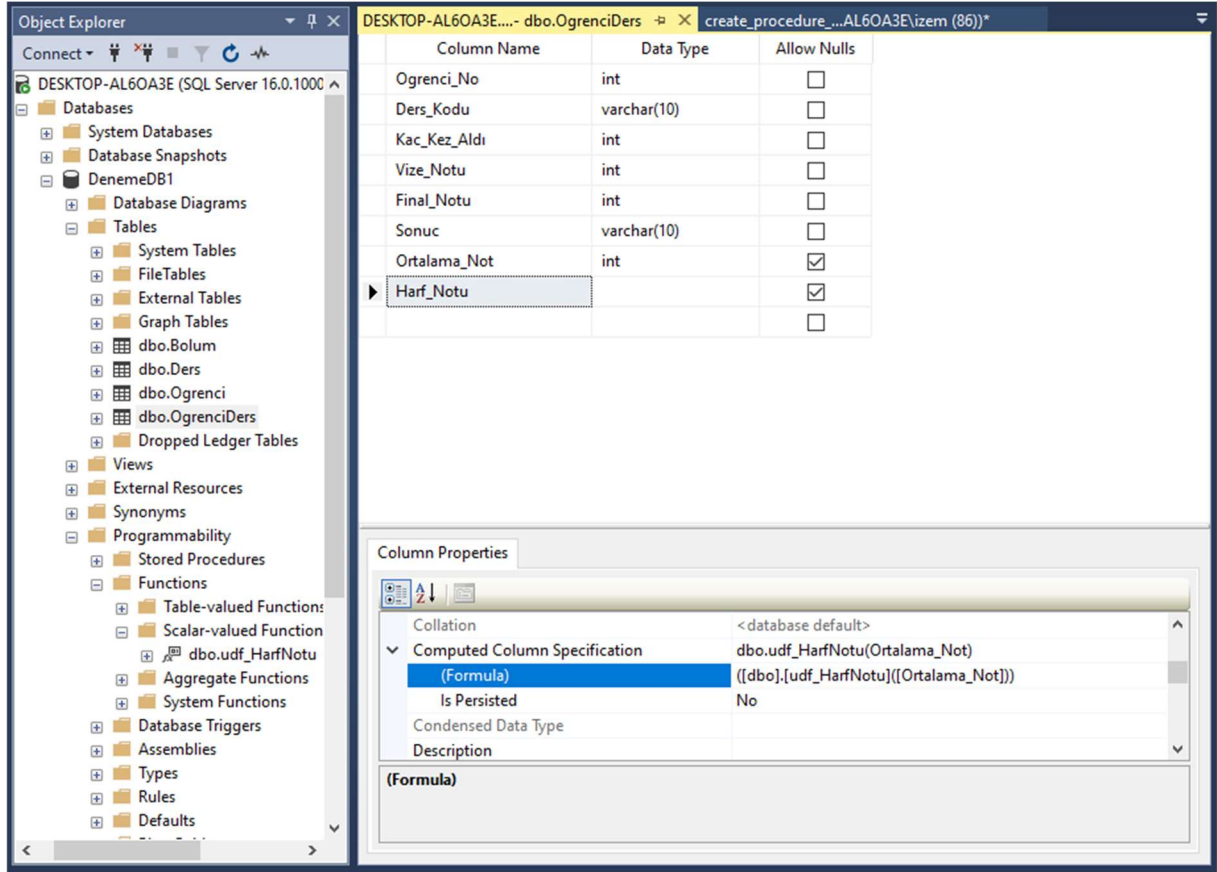
SELECT * FROM OgrenciDers
```

The results grid shows the following data:

Ogrenci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	Ortalama_Not	Harf_Notu
19	9	YAD02	1	75	100	Geçti	93
20	6	TBT01	1	43	61	Kaldı	56
21	3	YAD02	1	98	65	Geçti	75
22	14	MAT01	1	56	76	Geçti	70
23	1	KIM01	1	77	60	Geçti	65
24	3	MUH01	1	47	83	Geçti	72
25	1	YAD02	1	81	85	Geçti	84
26	5	HKG01	1	85	74	Geçti	77
27	14	AYH01	1	54	30	Kaldı	37
28	15	KIM01	1	76	53	Geçti	60
29	1	ATA01	1	34	61	Kaldı	53
30	7	YAD01	1	43	72	Geçti	63

The status bar at the bottom indicates 'Query execu...', 'DESKTOP-AL6OA3E (16.0 RTM)', 'DESKTOP-AL6OA3E\izem (80)', 'DenemeDB1', '00:00:00', and '30 rows'.

OgrenciDers tablosu, sağ tık, Design adımları izlenerek aşağıdaki ekrana ulaşıldı. Harf_Notu kolonu seçilerek, kolonun Computed Column Spesifikation özelliği altında yer alan Formula alanına, başında “dbo.” ifadesiyle birlikte, girdisi Ortalama_Not kolonu olacak şekilde tanımlanan fonksiyon girildi.



Aşağıdaki gösterildiği gibi OgrenciDers tablosu SELECT komutuyla çağrıldığında, Harf_Notu kolonunda yer alan tüm kayıtlar, ortalamaya karşılık gelen harf notları ile değiştirilmiş oldu.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'DESKTOP-AL6OA3E (SQL Server 16.0.1000)'. The main query window is titled 'create_procedure_...AL6OA3E\izem (86)*' and contains the following SQL script:

```
--ALTER TABLE OgrenciDers
--ADD Ortalama_Not int, Harf_Notu varchar(10);
--DROP COLUMN Harf_Notu, Ortalama_Not;

--UPDATE OgrenciDers
--SET Ortalama_Not =CAST(ROUND((Vize_Notu)*0.3 + (Final_Notu)*0.7, 0) AS int)

SELECT * FROM OgrenciDers
```

The Results pane shows a table with 9 columns: Ogranci_No, Ders_Kodu, Kac_Kez_Aldi, Vize_Notu, Final_Notu, Sonuc, Ortalama_Not, and Harf_Notu. The data is as follows:

Ogranci_No	Ders_Kodu	Kac_Kez_Aldi	Vize_Notu	Final_Notu	Sonuc	Ortalama_Not	Harf_Notu
19	9	YAD02	1	75	100	Geçti	AA
20	6	TBT01	1	43	61	Kaldı	DC
21	3	YAD02	1	98	65	Geçti	CB
22	14	MAT01	1	56	76	Geçti	CB
23	1	KIM01	1	77	60	Geçti	CC
24	3	MUH01	1	47	83	Geçti	CB
25	1	YAD02	1	81	85	Geçti	BA
26	5	HKG01	1	85	74	Geçti	CB
27	14	AYH01	1	54	30	Kaldı	FF
28	15	KIM01	1	76	53	Geçti	CC
29	1	ATA01	1	34	61	Kaldı	DC
30	7	YAD01	1	43	72	Geçti	CC

The status bar at the bottom indicates 'Query execu... | DESKTOP-AL6OA3E (16.0 RTM) | DESKTOP-AL6OA3E\izem (80) | DenemeDB1 | 00:00:00 | 30 rows'.

Aşağıdaki ekran görüntüsünde ise oluşturulan fonksiyona girdi olarak ortalama not (örneğin 70) verilmek suretiyle, fonksiyonunun döndürdüğü harf notu gösterilmektedir.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'DESKTOP-AL6OA3E (SQL Server 16.0.1000)'. The main query window is titled 'create_procedure_...AL6OA3E\izem (86)*' and contains the following SQL script:

```
--ALTER TABLE OgrenciDers
--ADD Ortalama_Not int, Harf_Notu varchar(10);
--DROP COLUMN Harf_Notu, Ortalama_Not;

--UPDATE OgrenciDers
--SET Ortalama_Not =CAST(ROUND((Vize_Notu)*0.3 + (Final_Notu)*0.7, 0) AS int)

--SELECT * FROM OgrenciDers

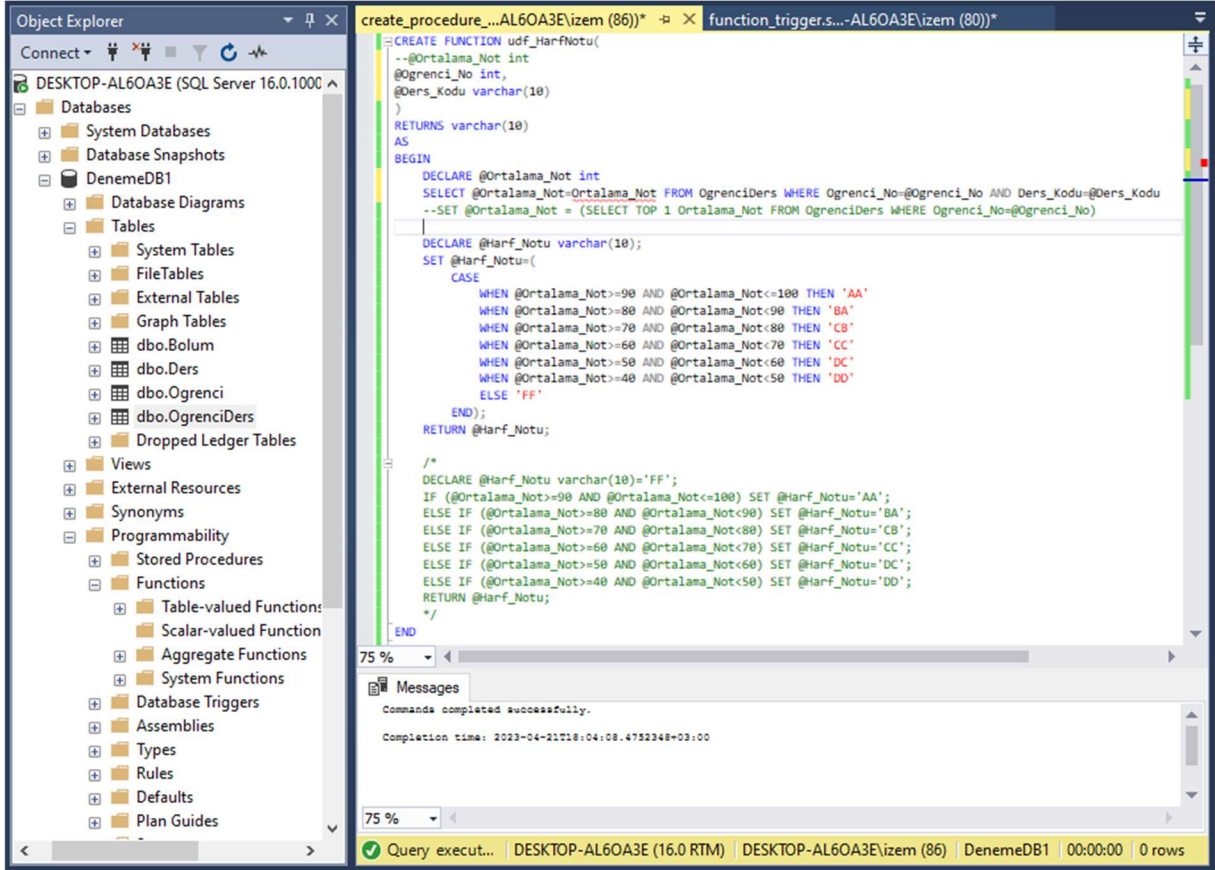
SELECT dbo.udf_HarfNotu(70)
```

The Results pane shows a table with 2 columns: (No column name) and Harf_Notu. The data is as follows:

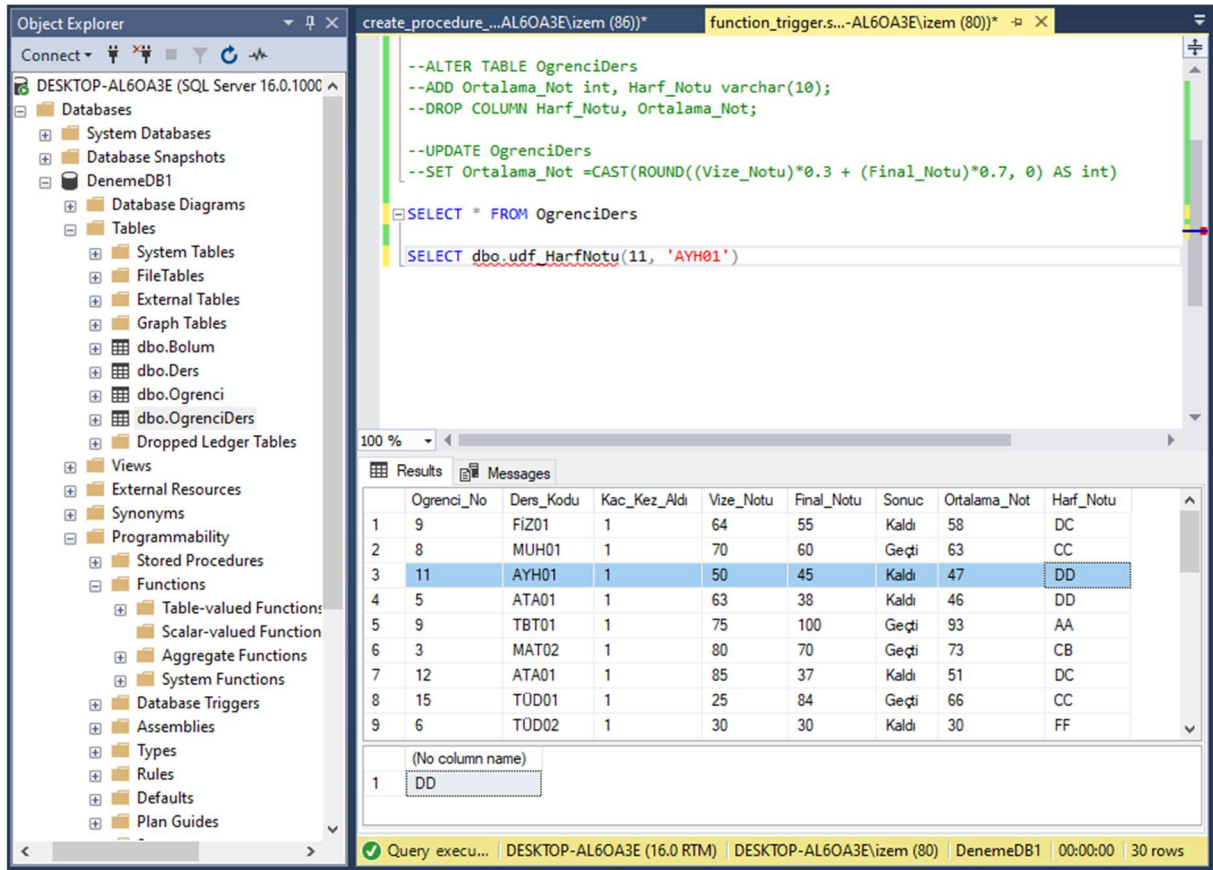
(No column name)	Harf_Notu
1	CB

The status bar at the bottom indicates 'Query execut... | DESKTOP-AL6OA3E (16.0 RTM) | DESKTOP-AL6OA3E\izem (80) | DenemeDB1 | 00:00:00 | 1 rows'.

Aşağıda gösterildiği gibi fonksiyonun girdisi öğrenci no. ve ders kodu olacak şekilde değiştirildi ve gerekli düzenlemeler yapıp fonksiyon yeniden oluşturuldu.



Yukarıdaki modifikasyonlar sonrası fonksiyona öğrenci no. ve ders kodu girdi olarak verilirse, ilgili öğrencinin söz konusu derse ilişkin harf notu aşağıda gösterildiği gibi elde edilir.



26. CREATE INDEX (3 puan)

CREATE INDEX index_name

ON table_name (column1, column2, ...);

Parametre olarak tablo adını, kolon isimlerini ve indeks ismini alan “CREATE INDEX” ifadesi, tabloları indekslemede kullanılır. Yukarıdaki sentaks, tekrarlayan değerlere izin verir. İndeksler, veri tabanından verilerin daha hızlı çağrılmasını sağlar. Kullanıcılar indeksleri göremez, onlar sadece aramaları ya da sorguları hızlandırır.

CREATE UNIQUE INDEX index_name

ON table_name (column1, column2, ...);

Yukarıdaki sentaks ise tekrarlayan değerlere izin vermez.

Aşağıdaki sorguda, OgrenciDers tablosunun Ogrenci_No kolonu kullanılarak idx_ogrenci_no isimli bir indeks oluşturulmuştur.

CREATE INDEX idx_ogrenci_no

ON OgrenciDers (Ogrenci_No);

